An Analysis of the Inspection of Mechanical Parts Using Dense Range Data

Edvaldo Marques Bispo

Ph.D.
University of Edinburgh
1996

Abstract

More than ever, efficiency and quality are key words in modern industry. This situation enhances the importance of quality control and creates a great demand for cheap and reliable automatic inspection systems. Taking into account these facts and the demand for systems able to inspect the final shape of machined parts, we decided to investigate the viability of automatic model-based inspection of mechanical parts using the dense range data produced by laser stripers.

Given a part to be inspected and a corresponding model of the part stored in the model data base, the first step of inspecting the part is the acquisition of data corresponding to the part, in our case this means the acquisition of a range image of it. In order to be able to compare the part image and its stored model, it is necessary to align the model with the range image of the part. This process, called registration, corresponds to finding the rigid transformation that superposes model and image. After the image and model are registered, the actual inspection uses the range image to verify if all the features predicted in the model are present and have the right pose and dimensions. Therefore, besides the acquisition of range images, the inspection of machined parts involves three main issues: modelling, registration and inspection diagnosis.

The application, for inspection purposes, of the main representational schemes for modelling solid objects is discussed and it is suggested the use of EDT models (see [Zeid 91]). A particular implementation of EDT models is presented.

A novel approach for the verification of tolerances during the inspection is proposed. The approach allows not only the inspection of the most common tolerances described in the tolerancing standards, but also the inspection of tolerances defined according to Requicha's theory of tolerancing (see [Requicha 83]). A model of the sensitivity and reliability of the inspection process based on the modelling of the errors during the inspection process is also proposed.

The importance of the accuracy of the registration in different inspections tasks is discussed. A modified version of the ICP algorithm (see [Besl & McKay 92]) for the registration of sculptured surfaces is proposed. The maximum accuracy of the ICP algorithm, as a function of the sensor errors and the number of matched points, is determined.

A novel method for the measurement and reconstruction of waviness errors on sculptured surfaces is proposed. The method makes use of the 2D Discrete Fourier Transform for the detection and reconstruction of the waviness error. A model of the sensitivity and reliability of the method is proposed.

The application of the methods proposed is illustrated using synthetic and real range images.

Acknowledgements

First of all, I would like to thank Robert Fisher who besides being a good supervisor was also a very good friend.

I would like to thank Andrew Fitzgibbon, David Wren, Dibio Borges, Giovanni Pettinaro, Maurizio Pilu, Pete Oliver and Mark Wright for their many suggestions, help with the writing up of the thesis and friendship.

I wish also thank my family that helped me to go through the bad times of these last four years.

Finally, I would like to thank the CNPq in Brazil which supported me financially during my PhD.

Declaration

I hereby declare that I composed this thesis entirely myself and that it describes my own research.

Contents

List of Figures

Chapter 1

Introduction

This thesis is concerned with the investigation of the use of dense range data for the automatic inspection of the shape of mechanical parts. In this chapter we present the motivations and objectives behind our work, and the research framework we used for investigating some of the main issues involved in the inspection problem.

Our motivations and objectives are discussed in Section 1.1. The acquisition of range data for inspection purposes is discussed in Section 1.2. Section 1.3 presents the research framework we used. The general structure of this thesis is described in Section 1.4. The main contributions of the thesis are listed in Section 1.5. A summary of the chapter is presented in Section 1.6.

1.1 Motivations and objectives

In the same way as human beings, modern manufacturers are also engaged in a daily and eternal struggle for surviving that only the companies able to deliver the best quality at the lowest prices have a chance to win. More than ever, efficiency and quality are key words in modern industry. This situation enhances the importance of quality control and creates a great demand for cheap and reliable automatic inspection systems able to replace human inspection, which is not ideal for many reasons:

- The human inspection performance is subject to fluctuations due to fatigue and tedium.

- The necessity of performing inspection in hazardous environments in many circumstances.
- The high throughput and accuracy required in some situations, as in the case of hot steel slabs [Suresh *et al.* 83], that make the use of human inspectors completely impossible.

Taking into account these facts and the demand for systems able to inspect the final shape of machined parts, we decided to investigate the viability of automatic model-based inspection of mechanical parts using range data. The motivations for investigating the use of range data were many. First of all, we wanted to take advantage of the explicit information of shape conveyed in range images, but besides that we were also motivated by the existence of a great variety of commercial rangefinders (able to acquire range images fast and reliably) and some well known techniques for the processing of range images.

Clearly, the design and implementation of model-based inspections systems using dense range data presumes the study of some important issues:

- Which types of inspection are possible to carry out using range data ?
- What accuracy and precision of inspection are achievable given the accuracy and precision of the original range data ?
- What is the best way of modelling the objects for inspection using range data ?
- How to use the model of the object being inspected to extract the desired features from the range image with the necessary accuracy and precision ?

In this thesis we investigate these issues by proposing not only a series of inspection procedures, but also statistical models of the failure rates of these inspection procedures.

Our main objective is not the determination of the exact failure rates associated with each inspection procedure, but to show how it is possible to evaluate numerically the failure rates of inspection procedures.

Our concern with the determination of the failure rates comes from the fact that numerical determination of the failure rates is essential for the determination of the adequacy and final cost of the inspection procedures designed.

1.2 Why laser stripers ?

One of the most common ways of acquiring 3D information for inspection involves tactile sensing, using co-ordinate measuring machines (CMM) (see [Menq & Yau 92] for an example of an application). CMMs are able to produce very accurate and precise results but, despite their popularity, CMMs have some important drawbacks related to their slow rate of data capture and the difficulties of programming them for inspection tasks. Also, due to their very slow rate of capture, CMMs produce sparse range information that does not allow the evaluation of surface shape as whole. Therefore, because we intended to be able to make a complete evaluation of surface shape, we discarded the use of CMMs from the very beginning.

Among the non-contact techniques for producing range images the most important ones are (see [Besl 88a] and [Poussart & Laurendeau 88] for extensive reviews):

- *Radars*: based on the emission and reception of electromagnetic waves, the most common types are the time of flight, amplitude modulation and frequency modulation.
- *Shape from shading*: surface orientation is recovered using a model of surface reflectance (usually lambertian), knowledge of light source position and making the assumption of surface smoothness.
- *Photometric stereo*: uses knowledge about surface reflectance and multiple light source positions to recover surface orientation.
- *Passive triangulation*: pairs or triples of 2-D images obtained from different viewpoints are combined to determine the depth of the observed surfaces. The depth calculation is based on finding the correspondences between the 2-D images' features and then using these correspondences and geometric principles (triangulation).

- *Active triangulation*: emission of some form of structured light (*e.g.* stripes, coded binary patterns) and use of geometric principles for the calculation of depth. Depth is recovered using a camera model and knowledge about the geometry of the structured light.

- *Moiré techniques*: range measurements are obtained through the modulation and demodulation of light using special gratings, the working principle being based on the interference phenomena.

Radar techniques, in general, are very expensive and are applied to measure large depth values. Moiré techniques are extremely accurate, but are also very slow and involve quite a lot of processing to calculate the depth information. Passive triangulation techniques suffer the burden of establishing the correspondence between the 2-D images, which is a very complicated and ill-posed problem. Shape from shading and photometric stereo do not recover depth information explicitly and depend on a large number of assumptions about the environment.

Considering these facts and the difficulties involved on using CMMs, we decided to use a rangefinder based on the active triangulation principle. More specifically, we propose to investigate the use of a system like the Edinburgh's laser striper [Naidu *et al.* 90], which consists of a pair of cameras and a plane of laser light used to project stripes on the object. Laser stripers are compatible with conveyor belts and, therefore, easy to integrate to standard manufacturing processes. Another motivation for the use of laser stripers was the existence of commercial systems able to acquire dense range images with an accuracy near to 10 μm.

1.3 The general framework of research

Given a part to be inspected and a corresponding model of the part stored in the *model data base*, the first step of inspecting the part is the *acquisition of data* corresponding to the part. In our case, this means the acquisition of a range image of the part. In order to be able to compare the part image and its stored model, it is necessary to align the model with the range image of the part. This process, called *registration*, corresponds to finding the rigid transformation that superposes model and image. After the image

and model are registered, the actual *inspection* uses the range image to verify if all the features predicted in the model are present and have the right pose and dimensions.

In many cases not all the features to be inspected are visible in one view of the object, and it is necessary to acquire other images of the part being inspected. Therefore, an important element in the inspection process is the *planner*, which takes account of the current position of the part (known after the first registration), the model of the shape of the part, the model of the sensors, and the list of features to be inspected to guide the acquisition of images corresponding to different views of the object.

Most of the time the *model data base* will be built using information about the shape of ideal parts and the tolerances associated with these parts supplied by the part manufacturer. However, there may be situations in which this information is not available and one has just physical examples of good and bad parts. In these cases, an interesting characteristic of an ideal system would be its ability to *learn models* of the part to be inspected from the images of the good and bad parts (that are used to determine the allowable variations around the nominal shape).

From all the considerations above, we can envisage the diagram of a general framework for research, given in Figure 1.1, composed by the following modules: *data acquisition, model data base, registration, inspection, planner* and *model learning.* Note that the modules in dashed lines are not part of the main objectives of this research.

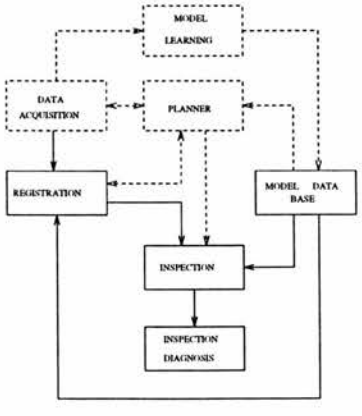


Figure 1.1: General system diagram.

1.4 The thesis structure

We start this thesis by giving a general account of the research done in machine vision inspection and some of the main issues related to it in Chapter 2. Chapter 3 focuses on the modelling problem and describes our particular approach to it. Chapters 4 presents a new approach for carrying out the inspection task. The problem of registering free-form surfaces is discussed in Chapter 5. The detection of waviness errors on sculptured surfaces is discussed in chapter 6. Finally, Chapter 7 presents our final conclusions and comments. The general structure of this thesis is described in the following:

- **Chapter 2 - Literature Review:** Literature review of some of the main research in machine vision inspection and the main issues related to the inspection problem (modelling, registration and data acquisition).
- **Chapter 3 - Model Data Base:** Discussion about the problem of modelling machined parts for inspection and the approach we adopted for solving it.
- **Chapter 4 - Model Based Inspection:** Implementation of some basic inspection tasks according to the standards and Requicha's theory of tolerancing [Requicha 83].
- **Chapter 5 - Registration:** The results of our experiments with the registration of free-form surfaces using a modified version of the ICP (Iterative Closest Point) algorithm [Besl & McKay 92].
- **Chapter 6 - Periodic Deformation Detection and Waviness Measurement:** Development of a specific method for the detection of periodic deformation in free-form surfaces and waviness error measurement.
- **Chapter 7 - Conclusions:** Conclusions and final comments on the overall results.

1.5 Contributions

This thesis presents many contributions to the model based inspection of mechanical parts using dense range data. We concentrate on the issues of modelling, registration

and inspection diagnosis and:

1. Adopted the use of EDT models [Zeid 91] for representing tolerance syntax and presented a particular implementation of EDT model that accommodates the representation of tolerances according to Requicha's theory of tolerance [Requicha 83].

2. Proposed a novel approach (task oriented) for the inspection of the most common tolerances described in the standards and in Requicha's theory of tolerance. The approach allows the numerical evaluation of the inspection procedures proposed, as well as it defines a procedural semantics to the EDT models.

3. Discussed the use of the ICP [Besl & McKay 92] for inspection purposes and proposed a multiscale modified version of the algorithm. The limits in the accuracy of the registration using the algorithm were also analysed.

4. A novel method for the detection, reconstruction and measurement of waviness errors based on the use of DFT was proposed and a model for the numerical evaluation of the quality of the results was developed.

1.6 Summary

In this chapter we have discussed our motivations for investigating the inspection of machined parts using dense range data, the acquisition of range data for inspection purposes and our general framework of research. The general structure of this thesis and its main contributions were also discussed.

Chapter 2

Literature Review

In this chapter we present a brief account of the most important research related to the inspection problem. We start the chapter by giving an overview of the research done towards the construction of automatic inspection systems in Section 2.1. After that we present a brief literature review of the main topics related to our research. The acquisition of range data is discussed in Section 2.2. Section 2.3 reviews registration techniques. Modelling and tolerancing issues are reviewed in Sections 2.4 and 2.5. Section 2.6 summarises the chapter.

2.1 Automated vision inspection

In this section we describe the main application areas of automated vision inspection and then review the work done on the inspection of mechanical parts using range information.

There is a considerable amount of research about applications of machine vision to automatic inspection. Some of the most important applications are listed below.

- **Printed wiring boards:** the inspection of printed wiring boards demands the verification of very complicated patterns at a very high throughput rate, with a high probability of defect detection and a low false alarm rate. Examples of the work done in this field can be seen in [Pau 82], [Hara *et al.* 83], [Silven *et al.* 88], [Fujihara *et al.* 94] and [Hara *et al.* 95].

- **Wafers:** wafer inspection involves wafer flatness measurement, wafer surface and probe mask inspection, and critical dimension measurement. More details and examples of applications can be seen in [Pau 82], [Jain *et al.* 86], [Yoda *et al.* 88], [Khalaj *et al.* 94] and [Hild *et al.* 95].

- **Mechanical parts:** a typical example of this kind of application is the use of X-rays for the detection and measurement of internal structures of metallic objects, *e.g.* [Hendengren 88] and [Boerner & Strecker 88].

- **Surface inspection:** examples of applications in this area are the detection and classification of surface imperfections occurring in cast hot steel slabs in a steel mill [Suresh *et al.* 83], the identification and localisation of defects in wood [Conners *et al.* 83], the quality control of the surface of glass [Wilder 86], the inspection of specular surfaces [Sanderson *et al.* 88], the inspection of billets, bars and wire rods [Kosakai *et al.* 93] and the detection of corrosion in aircraft structures [Komorowski *et al.* 96].

- **Solder joints:** Solder joints of printed circuit boards have defects such as excess of solder, shortage of solder in some areas and blow holes, all of which cannot be detected by electric circuit. The visual inspection of solder joints is a very difficult problem due to the great variability in the appearance of acceptable solder joints and the highly specular nature of flowed solder surfaces. Specific applications can be seen in [Nakagawa & Ninomiya 87], [Bartlett *et al.* 88], [Hiroi *et al.* 93] and [Pierce *et al.* 94].

Despite the many publications about automatic inspection, to the best of our knowledge, there are only three recent published works about the use of dense range data image obtained through active triangulation for inspection purposes [Marshall 89], [Marshall *et al.* 91] and [Newman & Jain 95]. Section 2.1.1 reviews these works.

The only other relevant research about the use of range data for inspection is concerned with the use of the sparse data produced by coordinate measuring machines (CMM). The use of sparse range data produced by coordinate measuring machines (CMM) is reviewed in section 2.1.2. This section is not a review on the use of CMMs as a

measuring apparatus, but instead focuses on the research about the processing of the sparse range data for the model based inspection of the shape of complex surfaces.

2.1.1 Inspection of mechanical parts using dense range data

[Marshall 89] and [Marshall *et al.* 91] proposed a model based inspection system using dense range data acquired using a laser striper (see Section 2.2 for more details on laser stripers). In his system the original image is initially segmented and then matched with the part's model using an approach originally proposed by Faugeras (see Section 2.3 for more details on Faugeras' technique). After the alignment between model and data, the inspection of the part takes place. During the inspection the dimensions and position of all the visible features are checked against their tolerances.

A region growing algorithm was used for segmenting the range image [Page 88]. An accuracy the order of a tenth of millimeter on the depth and of one degree in the normal angle is reported in the segmentation.

Marshall implemented some improvements to Faugeras' technique to improve its accuracy and efficiency. After improvement, an accuracy of less than one tenth of a millimeter in the translation is reported. Comparative studies between Faugeras' technique and Grimson's technique for the matching stage of the inspection were also presented. The studies concluded that Faugeras' technique was not only the most accurate one but also the fastest and the most robust to noise

Marshall also discussed the extension of his method to objects that do not have planar patches, but he has only presented results for surfaces with enough visible planar patches to allow the alignment between model and data.

In his work Marshall concentrated on the reduction of the errors during the registration phase of the inspection process. Despite his clear concern with the errors, Marshall never tried to establish a formal model of the errors during the whole inspection process. This fact made it impossible to him to develop formal models of the inspection process, or to evaluate numerically the quality of the inspection processes he developed. Furthermore, Marshall never discussed the inspection of sculptured surfaces.

[Newman & Jain 95] proposed a system for 3D CAD-based inspection of castings using

range images. In [Newman & Jain 95] system the castings are inspected to detect gross defects and the following specific defect conditions:

- unwanted surface defects (*e.g.* pits),
- insufficient and excess of material conditions,
- design feature defects such as circularity and cylindricity defects and
- wrong feature dimensions.

[Newman & Jain 95] proposed three different techniques for the inspection of the castings:

1. **Gross defect detection:**

   Gross defects in the range images of castings are inspected using bidirectional template matching. The template matchings are used to check assembly integrity and the presence or absence of gross manufacturing features, such as slots of large holes.

   Three different families of matching statistics are proposed to detect the gross defects in the range image.

2. **Surface shape inspection:**

   [Newman & Jain 95] proposed a method for the inspection of planarity and cylindricity of the planar and cylindric surfaces of castings. The inspection process starts with the segmentation of the range image and the identification of the planar and cylindrical surfaces.

   The planar surfaces are inspected by comparing their deviations to a plane fitted to the range image. Any set of two or more connected pixels that exceeds the threshold for pit defects is classified as a pit defect. Cylindrical surfaces are inspected in a similar way.

3. **Feature inspection and dimensional tolerance:**

   [Newman & Jain 95] developed a technique for the inspection of through-holes on the surface of the castings. Their technique inspects for the presence/absence of expected holes, detects undesired holes and inspects the shape and size of

through-holes.

Feature inspection assumes that the correspondence between the object surface in the image and the model surfaces in the CAD has been established. Candidate through-holes are found in the image by locating all voids in the range image (regions with no depth value). Voids consisting of only one pixel are ignored. The size of the holes are estimated using the number of pixels in the void region. Unexpected void regions that do not belong to the background are identified as pits defects.

[Newman & Jain 95] also developed techniques for the inspection of tolerance on circularity, distance between circular arcs, distance between planes and parallel planes.

All the techniques developed were tested using range images of castings. In the tests, the system of [Newman & Jain 95] was able to classify correctly 38 of 39 images of defective planar-surface castings and 7 of 8 images of defective cylindrical-surface castings. With the defect-free castings 5 of 5 images of cylindrical castings and 4 of 6 images of planar-surface castings were classified correctly.

In the same way as [Marshall *et al.* 91], [Newman & Jain 95] were not concerned with the formal definition of the inspection procedures or with the numerical evaluation of the inspection procedures they developed. Furthermore, [Newman & Jain 95] also did not address the inspection of sculptured surfaces.

In this thesis, we try to extend the work of [Marshall *et al.* 91] and [Newman & Jain 95] through a careful modelling of the errors in all the stages of the inspection process with the objective of making it possible to evaluate numerically the quality of the inspection procedures developed. Besides that, we give special attention to the inspection of sculptured surfaces that we discuss in Chapters 5 and 6.

2.1.2 Inspection of mechanical parts using sparse range data

The use of CMMs for dimensional inspection is widespread in the industrial environment. Typical applications are in the dimensional measurement of dies, molds, gears, turbine blades, etc. [Sterki & Giinter 82], [Takagi *et al.* 86] and [Chambers & Brown 86]

are some illustrative research in the area.

[Sahoo & Menq 91] developed an algorithm for locating 3-D objects having complex and sculptured surfaces. The algorithm determines the pose of an object by minimising the sum of square distances between points in the surface model and measured points with respect to the parameters of the pose transformation. The algorithm is very similar to the ICP (see Section 2.3.4) in concept. [Sahoo & Menq 91] also discussed the singularity and accuracy of his algorithm according to the shape of the model object.

[Menq & Yau 92] presented an improvement of [Sahoo & Menq 91] in which the estimation of pose does not involve the solution of complicated nonlinear equations, but the calculation of a pseudoinverse of the transformation matrix. [Menq & Yau 92] also developed a statistical model to determine the minimum number of required measurement points. The model assumed a normal distribution of the manufacturing process and an acceptable failure rate. Results were shown for an experimental part manufactured by a CNC (coordinate numerical controlled) machine.

[Menq *et al.* 92] and [Yau & Menq 92] discuss a structure of an intelligent planning environment for automated dimensional inspection using coordinate measuring machines. The system proposed is composed of 5 modules:

- **inspection specification module:** generation of an inspection specification from inspection attributes.
- **inspection planning:** creation of an initial inspection plan given the inspection specification.
- **verification module:** checking of the inspection plan to avoid interference between inspection path and part surfaces.
- **execution module:** data acquisition and part localisation using the traditional 3-2-1 method [Yau & Menq 92] or as in [Menq & Yau 92].
- **comparative analysis:** analysis of measurements against specified dimensions and tolerances using nonlinear least square for estimating deviation and statistical analysis for the detection of deterministic deformations in the deviations.

[Yau & Menq 93] discuss the designing of sculptured surfaces by minimising the total cost involved in the machining and inspection of the surfaces. Numerical controlled machines (CNC) are assumed to be used for the milling and CMMs for the inspection. The inspection costs are estimated assuming the sampling technique proposed in [Menq & Yau 92].

A detailed description of the statistical tests suggested in [Menq *et al.* 92] and [Yau & Menq 92] is presented in [Yang & Menq 93]. The paper discusses the use of Moran's statistic for the detection of deterministic deformations using the deviations between the measurements obtained using a CMM and the surface's model. The estimation of the deterministic determination through the fitting of B-splines is also suggested.

[Medland *et al.* 93] proposed an inspection procedure using CMMs in which grids of points on the feature being inspected are used for the inspection. The approach was successfully implemented in an experimental programme.

[Klages & Wilson 94] developed a prototype system to automate the programming of CMMs. The system allow design engineers, inspection experts and CMM programmers to work in an integrated and automated environment to produce the CMM programmes.

The applications of CMMs with contact prober to the dimensional inspection of manufactured products are restricted to the parts made of hard material. [Cheng & Menq 95] extended the applications of the CMMs to the inspection of objects made of soft materials (such as foils, plastic, wood wax and clay materials) by replacing the contact probes with inexpensive laser sensors.

[Yau & Menq 95] proposed an hierarchical planning system for path planning in dimensional inspection using CMMs. The system aims to automate the planning of collision-free inspection path for dies and molds. Simulation and experimental results are shown to illustrate the feasibility of automatic path planning.

TECHNIQUE	KIND	WORKING PRINCIPLE
shape from shading	passive	modelling of light transmission
photometric stereo	passive	modelling of light transmission
passive stereo	passive	triangulation and feature correspondence
radar	active	emission and reception of electromagnetic waves
Moiré	active	interference phenomena
focusing	active	Gauss thin lens law
active triangulation	active	structured light projection and triangulation

Table 2.1: Rangefinding techniques.

2.2 Data acquisition

This section reviews some of the most important techniques for 3-D data acquisition for industrial computer vision. Our objective is to illustrate the different techniques and describe their main characteristics rather than to produce an exhaustive review. More extensive reviews can be found in [Besl 88a], [Aggarwal & Chien 88] and [Poussart & Laurendeau 88], from which this section draws heavily, or in [Wagner 86].

Roughly speaking, we can divide the current rangefinding techniques in two basic classes: active or passive. In the active case a special device illuminates the scene, while in the passive case the scene illumination is provided by ambient lighting. Passive techniques are usually used in applications where the use of light sources is forbidden for security or environmental reasons. Active sensing can be used in a wide variety of applications as shape acquisition, inspection, navigation, etc. .

Table 2.1 enumerates the most used rangefinder techniques in industrial computer vision. Sections 2.2.1 to 2.2.6 review these techniques.

2.2.1 Shape from shading and photometric stereo

Shape from shading and photometric stereo techniques try to recover surface orientation from image brightness. The brightness of the image of a 3-D object depends on the object's shape, its reflectance properties, the distribution of the light sources and the sensing device. If all these elements are modelled it is possible to obtain an explicit equation (the reflectance map - see [Horn 77]) relating surface orientation and brightness. In general the reflectance map does not provide enough information to recover the surface orientation and additional constraints are needed.

[Horn 75] was the first to propose a solution for the shape from shading problem based

on the solution of a non-linear first-order partial differential equation.

[Brooks 79], [Strat 79] and [Woodham 79] used relaxation methods for recovering surface shape from the information in the reflectance maps.

[Bruss 81] was the first to address the problem of recovering shape when there is no *a priori* knowledge of the scene. In these cases it is not possible to make global assumptions about the scene and local shape from shading analysis is necessary. [Pentland 84] presented a detailed discussion of the limitations of the shape from local shading analysis.

[Horn & Brooks 90] proposed an algorithm for the simultaneous recovery of surface orientation and height field. Brightness, gradient and integrability constraints are combined in a cost function to be minimised in order to recover shape. A more efficient version of Horn's algorithm was implemented by [Szeliski 91]. Experiments in two test images showed a considerable performance improvement in the new implementation of the algorithm.

The additional constraints necessary for recovering surface orientation using reflectance maps are obtained in photometric stereo techniques by correlating several distinct irradiance images obtained from the same viewpoint but with different illumination conditions. Shape is recovered by finding the surface orientation that satisfies the constraints imposed by the different reflectance maps corresponding to each illumination condition. [Woodham 79] was the first to propose the photometric stereo technique.

[Byungil & Burger 91] developed method for the independent calculation of local position and orientation of lambertian surfaces using photometric stereo. Experiments in test image show a relative accuracy of *15* % with a data acquisition rate of one point per second.

[Schultz 94] presented an algorithm for the recovery of the shape of specular surfaces using photogrammetric stereo. Experiments in controlled conditions show good agreement between the calculated and expected shape.

[Jones & Taylor 94] developed a robust algorithm for shape from shading using a new paradigm based on scale space surface.

An algorithm for extracting shape assuming perspective projection was developed by [Lee & Kuo 93]. The algorithm divides the image into a set of overlapping triangular domains, and reconstructs surface heights directly. Results with synthetic and real images are shown. [Lee & Kuo 94] generalises the idea of triangular patches and reflectance map linearisation to implement a multiple image algorithm for improving the accuracy of the surface reconstruction.

[Iwahory *et al.* 94b] used three images acquired under different light directions and a neural network for recovering shape. [Iwahory *et al.* 94a] presented a method for recovering the shape of lambertian surfaces using four images and an actively controlled point of light.

The recovery of shape from coloured images is discussed by [Font *et al.* 92], [Petrov 93] and [Kontsevich *et al.* 94].

From all that it was exposed, we can see that the shape from shading and photometric stereo methods still depend on a large number of assumptions and are not robust enough to be used for inspection purposes in the manufacturing environment.

2.2.2 Passive stereo

Passive stereo techniques combine pairs or triples of 2-D images obtained from different viewpoints to recover depth. Depth calculation is done by finding corresponding features in the 2-D images and then using triangulation between the matched features.

The main difficulty involved with the passive stereo techniques is to establish the correspondence between the features in the 2-D images, *i.e.* the correspondence problem. A computer vision theory of the stereopsis process in humans was first proposed by [Marr & Poggio 79]. According to [Marr & Poggio 79], the correspondence problem is solved by filtering the original images using the Laplacian of a Gaussian function, finding the zero-crossings in the filtered image and matching the zero-crossings in two different images. A coarse filtered image is first used to get a first estimate of disparity. Better estimates of disparity are obtained using finer filtered images.

This model of human stereopsis was implemented in [Grimson 81] and [Marr & Poggio 81] and presented a reasonable performance similar to the human subjects. [Grimson 84]

also implemented a modified version of this model including the notion of figural continuity. In the modified version a figural constraint was used for eliminating the random matches.

Besides the research addressing the stereo process in the human visual system, there is a great deal of work examining computational aspects of the stereo process for different applications.

[Baker 81] presented an algorithm for stereo matching in which connected edges are used as matching features and edges correspondences found to be wrong are cooperatively removed. This algorithm was later improved by adding an intensity correlation phase in which edge descriptions are correlated using local properties such as contrast, image slope and intensity - see [Baker & Binford 80].

An efficient algorithm for recovering depth was proposed by [Kim & Aggarwal 85] Their algorithm matches feature points (zero-crossing points) based on the vertical connectivity of their zero-crossing patterns. After that a relaxation process is used to reduce the ambiguities arising from the first phase. According to the authors, the initial matching of zero-crossing points allows a fast convergence of the relaxation process: after the fourth and seventh iteration 90% to 98% of the feature points are matched correctly.

An interesting solution to the stereopsis problem was proposed by [Boyer *et al.* 86]. In his approach, called structural stereopsis, first a structural description of the images being matched is built. This description is based on a set of primitives P and relations R. The primitives P correspond to segments extracted from a radial-valued skeleton obtained from the binary image. The relations R describe how the segments are linked. The correspondence problem is solved by matching the structural descriptions of the images. The algorithm was successfully used for sorting rods of different length (but identical diameter) automatically using a robot.

A stochastic optimisation approach was proposed by [Barnard 86]. The stereo matching problem was formulated as the minimisation of an energy cost function and solved using simulated annealing. The objective of the minimisation process is to find a disparity map such that matched points have similar intensities and the disparity map

is smooth. A similar approach was also used for [Terzopoulos *et al.* 87] in which the objective is to minimise an energy cost function that combines a similarity functional with a controlled continuity constraint. The energy function is first minimised in a coarse scale, that is later refined to avoid problems with local minima. Results with natural scenes are used to shown the approach performance.

A discussion of different techniques for measuring disparity is presented in [Michael *et al.* 91].

[Cochran & Medioni 92] presented an robust stereo vision system able to deal with outdoor scenes and man-made objects. Their system integrates area and feature based primitives to obtain accurate and dense disparity maps. Images of size 256×256 are processed in *5* hours for obtaining approximately *96* % of the matches correctly.

A multi-primitive hierarchical approach is proposed by [Marapanese & Trivedi 94] in which region, linear edge and edgel based stereo are integrated hierarchically. Experimental results show that the system can find from *85* % to *100* % of the matches correctly and presents an accuracy of approximately *2* parts in *100*.

[Kanade & Okutomi 94] presents an algorithm for adaptively selecting the window size for stereo matching based on sum of square differences. Results with real images show considerable improvement in performance.

Despite the great progress in the development of passive stereo systems, these systems still suffer of the burden of establishing the correspondences between pairs or triples of 3D images. This fact makes the systems based in the passive stereo less than ideal for inspection purposes because of the high throughput rate that is normally demanded for inspection.

2.2.3 Radar

When a signal (*e.g.* ultrasonic pulse or electromagnetic wave) is transmitted to an object; some of the energy of the signal is reflected back to the transmitters. Radars work under the assumption that the reflected signal can be detected and, therefore, the distance r_{dist} between the transmitter and the reflecting object can be calculated as a function of the speed of the signal propagation V and the time τ between the

REFERENCE	RESOLUTION (accuracy/depth of field)	DATA CAPTURE (points/s)	COMMENTS
[Lewis & Johnston 77]	20 mm / 3 m	100	used in the Mars rover mobile robot
[Heikkinen *et al.* 86] and [Ahola *et al.* 85]	20 mm / 1.5 m	10,000	working volume at 2.5 m
[Ross 78]	300 mm / 75 m	5 E+8	only patented
[Banic *et al.* 87]	300 mm / 45 m	20	used for airborne hydrographic surveying

Table 2.2: Pulsed radars.

REFERENCE	RESOLUTION (accuracy/depth of field)	DATA CAPTURE (points/s)	AMBIGUITY INTERVAL	COMMENTS
[Nitzan *et al.* 86]	2 mm / 11 m (SNR=67 db)	2 (SNR=67 db)	16.6 m (SNR=67 db)	created registered range and intensity images
[Zuk & Dell'eva 83]	61 mm / 9.750 m	32,768	?	developed in the Environ. Res. Inst. of Michigan
[Sampson 87]	20 mm / 19.5 m	128,863	19.5 m	as above uses 3 frequencies
[Svetkoff *et al.* 84]	0.1 mm / 0.2 m	100,000	?	working vol. at 2.3 m
[Binger & Harris 87]	18 mm / 7.2 m	31,250	9.4 m (Odetics)	commercially available
[Miller & Wagner 87]	25 mm / 6 m	1,000	?	uses a modulated infrared LED

Table 2.3: Amplitude modulated continuous wave radars.

transmission and detection of the signal by the radar:

$$r_{dist} = \frac{V \cdot \tau}{2} \tag{2.2.1}$$

Several different principles have been used to measure range. Pulsed or time-of-flight radars emit energy in short pulses, wait for the detection of the reflected signal and then estimate the range using a measurement of the time between transmission and reception. Continuous-wave radars emit energy continuously either as amplitude modulated signals (AM) or frequency modulated signals (FM). In the AM case range is obtained by measuring the phase difference between the transmitted and the received signal. Because relative phase differences can only be determined with module 2π, radar AM rangefinders present an ambiguity interval equal to half of the wavelength of the radar's signal. FM radars use beat frequencies to obtain the distance to the reflecting object.

Tables 2.2, 2.3 and 2.4 summarise the main characteristics of some significant pulsed radars, AM radars and FM radars respectively. The construction of these tables was based on the Besl's review about active range sensors [Besl 88a].

The use of radars for inspection purposes is not ideal due to the relative small measurements of depth involved in the inspection process and the high cost of the radar systems in general.

REFERENCE	RESOLUTION (accuracy/depth of field)	DATA CAPTURE (points/s)	COMMENTS
[Hersman *et al.* 87]	12 mm / 1 m	2,048	vision system radar
[Hersman *et al.* 87]	0.05 mm / 2.5 m	0.37	metrology system
[Beheim & K. 86]	2.7 mm / 0.5 m	29.3	accuracy vary with depth of field

Table 2.4: Frequency modulated continuous wave radars.

2.2.4 Moiré techniques

In Moiré systems a beam of light is spatially amplitude-modulated by making the beam pass through a grating with regularly spaced patterns. After that, the beam is projected on a smooth surface and the information about the shape of the surface is coded in the phase of the light beam. Finally, the shape information in the light beam's phase is recovered through demodulation. This demodulation process might be done using another grating with regular patterns or via computer software.

Moiré techniques are useful for measuring relative distance to surface points on a smooth surface that does not have discontinuities. Absolute values can be obtained if a reference distance is given. Another disadvantage of Moiré techniques is that the magnitude of the surface slope in relation to the sensor point of view cannot be very big.

In the classical projection Moiré systems, the light beam is modulated and demodulated using two precisely matched gratings (the first in front of the light source and the second in front of the camera). The biggest problems related to the construction of such systems are the calibration and the automatic analysis of the fringes produced by the modulation-demodulation process.

If the depth of field is small, the modulation and demodulation of the light beam can be done using a single large grating which eliminates the problem of constructing two exactly matched gratings. Examples of the use of this technique are discussed in [Duncan *et al.* 80] and [Cline *et al.* 84].

The demodulation step in Moiré systems can be done by software, provided that a reference image of a flat plane is digitised beforehand. [Idesawa *et al.* 77] and [Idesawa & Yatagai 84] were some of the first to work with this technique that is called single-frame Moiré with reference.

[Halioua & Srinivasan 87] patented a multiple-frame phase-shifted Moiré technique. This technique is similar to the single-frame one except that, after frame projection, the projector grating is shifted to produce a specific phase shift which increases the range accuracy in one order of magnitude. Using this technique [Srinivasan *et al.* 85] was able to obtain $0.1\ mm$ of range accuracy over a depth of field of $1\ m$. [Boehnlein & Harding 86] built a very fast implementation of this algorithm and was able to acquire a 256×256 image in about $10\ s$ with an accuracy of $11\ \mu m$ over a $64\ mm$ depth of field.

[Gu *et al.* 94] developed a one step phase shift 3D system for measuring surface profile. The system used only one frame of digital image to extract shape and achieve an accuracy of $\frac{\pi}{30}$ in the laboratory experiments.

The recovery of the 3D shape of diffusely reflecting surfaces was implemented by [Engelhardt 91] using grid projection with a small depth of focus and focus sensing. [Engelhardt 91] used the Moiré effect between the projection grid and the CCD array.

An intelligent system for length measurement composed of a linear CCD array, grating and micro-computer was proposed by [Chen *et al.* 92]. The system applied a new fine-division method of Moiré fringes to increase resolution.

[Matsumoto *et al.* 92] developed a resolution-variable Moiré topography for measuring 3D profiles. Moiré fringes were formed by projecting 2 sets of interference fringes of laser beams. Validation of the system was done by testing with a small object.

An experimental setup for an automated projection Moiré was recently implemented by [Cardenasgarcia *et al.* 94].

Despite their great accuracy, Moiré techniques are not still reliable enough for their use in inspection process. Besides that, the methods based on the Moiré technique suffer the drawback of not being able to cope with big magnitudes of surface slope (discontinuities).

2.2.5 Focusing

Due to the diffraction of the light passing through a finite aperture, images are always blurred. The radius of the blur is minimum when the point is in focus, *i.e.* it satisfies

the Gauss thin lenses equation. Focusing techniques measure range using this property of camera lenses.

[Rioux & Blais 86] used the radius of the blur of a grid of projected points to recover depth. They were able to achieve an accuracy of *1 mm* over a *100 mm* depth of field. [Rioux & Blais 86] also developed a second sensor using multistripe illumination. Lines out of focus were seen split and the splitting distances were related to the range. Using this sensor they were able to capture a 256×240 range image in less than one second with an accuracy of *1 mm* over *250 mm*.

An accuracy of *0.3 mm* over a depth of field of *150 mm* were obtained by [Kinoshita *et al.* 86] using a point range sensor based on the projection of a conical ring of light and focusing.

[Ahuja & Abbot 93] developed method for integrating stereo, camera vergence and lens focus to recover shape using active control of camera focus and orientation. The result was an active vision system that cooperatively interleaved image acquisition and surface estimation to obtain dense maps. An average error of *0.15* % over *2m* of depth of field was reported.

[Nayar *et al.* 95] developed a real time range sensor based on the principal of focus analyses. The sensor uses inexpensive off-the-shelf imaging and processing hardware and it is able to acquire 512×480 depth maps at a frequency of *30 Hz* (video frame rate) with an accuracy of *0.3* %.

The application of focusing for inspection purposes is very promising, and we can already found systems of very good accuracy and high data capture rate.

2.2.6 Active triangulation

Active triangulation systems measure range by emitting some form of structured light on the object being imaged and using triangulation. As the range measurement depends on emission and reception of structured light in two different viewpoints, active triangulation systems can have the problem of missing parts of the object when a point is not visible in both of the two viewpoint directions.

In this thesis we will assume the use of the active triangulation principle for obtaining

REFERENCE	RESOLUTION (accuracy/depth of field)	DATA CAPTURE (points/s)	COMMENTS
[Rioux 84]	0.4 mm / 1 m	32,678	vision system radar
[Servo-Robot 87]	0.05 mm / 0.6 m	3,000	Saturn radar working volume at 80 mm
[Servo-Robot 87]	0.3 mm / 1 m	3,000	Jupiter radar working volume at 0.1 m
[Hymarc 87]	0.25 mm / 0.5 m	3,000	working volume at 600 mm
[Juha & Donahue 87]	0.025 mm / 6.25e-3 m	10,000,000	commercially available
[Bickel *et al.* 85]	0.025 mm / 0.088 m	?	similar to [Rioux 84]
[Faugeras & Herbert 86]	?	?	one laser spot and two horizontal detectors (uses turning table)
[Corporation 87]	0.75 μm / 300 μm	15	PRS-30 radar commercially available
[Diffracto. 87]	2.5 μm / 0.002 m	200	Model 300 commercially available
[Gottwald & Berner 87]	50 μm / 3 m	7.5	
[Lorenz 84]	2.5 μm / 1 m	?	split-beam illumination and optimal estimation theory
[Selcom. 87]	2 to 128 μm / 0.008 to 0.512 m	16,000	commercially available series of optical probes
[Harding & Tait 86]	9 μm / 0.15 m	16,000 (for 256 points per scan)	
[Piptone & Marshall 83]	19 mm / 7.8 m	500	
[Haggren & Leikas 87]	?	1.5	accuracy better than one over 10,000 uses subpixel accuracy and 4 cameras)
[Komorya & Tani 91]	? ?	? ?	used in in doors navigation in corridors
[Leseberg 92]	?	?	sensor for small ranges illumination and detection together

Table 2.5: Active triangulation: point structured light.

the dense range information. More specifically, we will assume the use of systems as the Edinburgh's laser striper [Naidu *et al.* 90] for acquiring range information. Our choice was based in the high accuracy and data capture rate observed in many commercial systems.

Although laser stripers assume the use of lines as structured light, through the rest of this section we will discuss all the forms of structured light that can be used in systems based in the active triangulation principle.

Many different patterns of structured light have been used for active triangulation: points, lines, multiple points, multiple lines and coded binary patterns are the most common. More details about these different techniques are given in the next sections.

2.2.6.1 Points and Multiple Points

Some of the most significant rangefinder systems using point and multiple point structured light are summarised in Table 2.5 and Table 2.6.

REFERENCE	RESOLUTION (accuracy/depth of field)	DATA CAPTURE (points/s)	COMMENTS
[Kanade & Fuhrman 87]	0.1 mm / 1 m	200	use 18 LED light source
[Labuz & McVey 86]	?	?	tracks multiple points on a grid over the scene

Table 2.6: Active triangulation: multiple point structured light.

REFERENCE	RESOLUTION (accuracy/depth of field)	DATA CAPTURE (points/s)	COMMENTS
[Corp. 87]	0.5 mm / 2.4 m	3,000	100X White Scanner (commercially available)
[Ozeki *et al.* 86]	20 mm / 0.50 m	1,176	60 cm x 60 cm field of view
[Silvaggi *et al.* 86]	0.25 mm / 0.05 m	?	working volume at 100 mm
[Corporation 87]	0.75 μm / 300 μm	320	LRS-30-500 commercially available
[Landman & Robertson 86]	8 μm / 0.0127 m	28	Eyecrometer (Octeck) commercially available
[Diffracto. 87]	50 μm / 0.019 m	256	Z-sensor commercially available
[Hamilton 96]	50 μm / ?	1000	commercially available
[Hamilton 96]	10 μm / ?	1000	commercially available
[Hamilton 96]	20 μm / ?	1000	commercially available
[Hamilton 96]	40 μm / ?	1000	commercially available

Table 2.7: Active triangulation: line structured light.

2.2.6.2 Line and Multiple Line

Many researchers used light striping for recovering depth information: [Shirai 71], [Will & Pennington 84], [Nevatia & Binford 73], [Rocker 74] and [Popplestone *et al.* 75] were some of the pioneers.

Table 2.7 and 2.8 illustrate some of the most recent work using laser stripers.

2.2.6.3 Coded binary patterns

The time acquisition in the active triangulation systems can be reduced when space encoding is used. The projection of grey-coded patterns makes it possible to encode 2^n wedge-shaped regions when n coded masks are projected.

[Solid Photography 77] was the first to use this principle for recovering depth information. [Potsdamer & Altschuler 82], [Inokuchi *et al.* 84] and [Yamamoto *et al.* 86] are some of the researchers that used this principle for range imaging.

REFERENCE	RESOLUTION (accuracy/depth of field)	DATA CAPTURE (points/s)	COMMENTS
[Jalkio *et al.* 85]	0.25 mm / 0.025 m	137	field of view 60 mm x 60 mm
[Mundy & Porter 87]	25 μm / 50 μm	16,000	tolerates 10 to 1 changes in reflectance

Table 2.8: Active triangulation: multiple line structured light.

REFERENCE	RESOLUTION (accuracy/depth of field)	DATA CAPTURE (points/s)	COMMENTS
[Solid Photography 77]	0.75 mm / 0.3 m	64,000	data acquisition for producing people's bust
[Rosenfeld & Tsikos 86]	2 mm / 0.05 m	93,600	10 grey code patterns
[Vuylsteke & Oosterlinck 86]	? / ? (7 bits accuracy)	58	64 × 64 range images from 604 × 576 intensity images
[Urquhart *et al.* 93]	0.5 mm / 0.3 m	?	accuracy obtained projecting 7 coded frames

Table 2.9: Active triangulation: coded binary patterns.

[Boyer & Kak 87] developed a light striping technique that uses just one image frame from a colour video camera. In their work colour was used for coding the space information.

Table 2.9 illustrates the performance of some significant systems.

2.2.6.4 Other forms of structured light

Besides points (multiple points), lines (multiple lines) and coded binary patterns, many other forms of structured light have been used for recovering depth.

[Pelowski 86] and [Nakagawa & Ninomiya 87] used crosses. [Wei & Gini 83] used circles. Grids were used by [Will & Pennington 84], [Hall *et al.* 87] and [Wang & Aggarwal 86]. [Asada *et al.* 86] used thick stripes. [Schewe & Forstner 86] reports a range sensor with *0.1 mm* accuracy over a depth of field of one meter, able to acquire approximately a point per second using random texture. [Urquhart *et al.* 93] reports an accuracy of *0.1* pixel and acquisition times of approximately one minute to acquire an image of *512* × *512* pixels using random textures.

2.3 Registration

The model based inspection of mechanical parts consists basically in a comparison between the model of the part being inspected and a range image of it. Therefore, before the inspection can take place, it is necessary to register or superpose the model of the part and the range image.

Given two different views of an object, the registration problem can be defined as the problem of finding the rigid spatial transformation that aligns these two different views.

Due to its importance in computer vision, the registration problem has been the object of study of many researchers. In a very rough way, the many techniques proposed to solve it can be divided into two principal classes:

1. **Matching techniques:** in which the solution of the registration problem is based on the explicit solution of the correspondences between the different views or model and data being registered.
2. **Optimisation techniques:** in which the solution of the registration problem is based on the minimisation of a suitable cost function.

The main drawback of the first class of techniques is related with the difficulties associated with finding the correspondences between the views or model and data being registered. This means that these techniques either demand searches of high complexity, or rely on the segmentation of the original image or demand a pre-processing stage in which geometric invariants are calculated for reducing the search complexity (geometric indexing).

The second class of techniques presents the advantage of not depending on the explicit resolution of the correspondence problem. However, due to their dependence on the minimisation of a cost function, these techniques depend on the initial estimate of the registration solution and often converge to a local minimum. This fact demands the use of many initial guesses to achieve a reasonable degree of reliability, which makes these techniques also very computationally expensive.

In general the matching techniques require more complex features such as surfaces, splashes [Stein 92] or Frenet frames [Gueziec & Ayache 94] while the optimisation techniques make use of the raw range data.

Concerning the use of the registration techniques for inspection purposes, the matching techniques are more indicated in the situations in which we have distinctive features that can guide the registration process. An excellent discussion of the registration for inspection purposes in this case can be seen in [Marshall 89].

In the case of sculptured surfaces, in which we can count only with the raw 3D data for obtaining the registration, the optimisation techniques are the best option. Because

REFERENCE	PROBLEM	DATA	OBJECT REPRESENTATION	DESCRIPTION
[Potmesil 83]	modelling	range image (hierarchical quadtrees)	network of bicubic parametric patches	search for maximising shape similarity
[Grimson & Lozano-Perez 84]	recognition	oriented surface points (range data)	geometrical description	search in IT pruned using local geometrical constraints
[Faugeras & Herbert 86]	recognition	segmented range image	sets of surfaces	search pruned using the estimated registration
[Arun *et al.* 87]	registration	sets of points	sets of points	use of SVD to register sets of corresponding points
[Ponce *et al.* 92]	registration (recognition)	range image, intensity image, image contours, orientation and curvature	parameterised surfaces (algebraic equations)	manipulation of algebraic equations (Elimination Theory)
[Besl & McKay 92]	registration	range image	point sets, polylines, parametric and implicit curves, faceted surfaces, parametric and implicit surfaces	minimisation of distance between set of points and a geometric entity (ICP algorithm)
[Chen & Medioni 92]	modelling	range image	range image	minimisation of distance between points in one surface and tangent planes of another
[Stein 92]	recognition	range image and intensity image	geometric indexing	uses invariants for matching
[Zhang 93]	registration	3D points	3D points	similar to ICP
[Gueziec & Ayache 94]	recognition	3D points	characteristic curves	estimation of torsion and curvature for matching

Table 2.10: Registration techniques.

of the problem of convergence to local minima, the optimisation techniques must be used together with jigs or mechanical fixtures to constrain the initial pose of the object being inspected.

Table 2.10 summarises some of the main registration techniques, and a more detailed description of these techniques is presented in Sections 2.3.1 to 2.3.10. As the solution of the registration problem is directly related to the recognition of objects in artificial vision systems, many works present the solution of the registration problem together with the description of whole systems. Every time this happened, we opted for giving a general picture of the work described.

2.3.1 Potmesil

[Potmesil 83] proposed a method for constructing surface models of arbitrarily shaped solid objects by matching the 3D surface segments describing these objects from different views. The surface models were composed of networks of bicubic parametric patches.

The surface information was obtained by photogrammetric and image-processing techniques from multiple images of an object. The surface patches obtained were converted into a hierarchical quadtree representation, in which the bottom level of the tree contained the original patches and the higher levels corresponded to merging of the original patches.

In the context of Potmesil's work, finding the match between 3D surfaces corresponded to finding a rigid 3D geometrical transformation which aligned the two surface segments. To calculate this rigid transformation Potmesil used a heuristic search algorithm that tried to minimise the orientation and shape differences between the shapes being registered.

The orientation and shape differences were measured at a number of evaluation points by using a ray-casting procedure. This consisted of considering a ray in the direction of the surface normal vector at an evaluation point and calculating:

1. **positional difference:** 3-D Euclidean distance from the origin of the ray to the nearest point on the other surface,
2. **orientation difference:** the angular difference of the surface normals at the two points defined above and
3. **curvature difference:** the magnitude of the difference of the surface curvatures at the points defined above

The selection of the evaluation points was done at each level of the quadtree representation by choosing as patch control points the surface points with maximum curvature.

The initial estimates of the match were calculated by aligning the normals in the top levels of the quadtrees.

2.3.2 Grimson and Lozano-Pérez

[Grimson & Lozano-Perez 84] proposed a methodology for identification and localisation of polyhedral objects using a finite set of measurements $\{\mathbf{s_i} = (\mathbf{p_i}, \mathbf{n_i})\}_{i=1,m}$ where:

- $\mathbf{p_i}$ is a point belonging to the ith object surface and
- $\mathbf{n_i}$ is the surface normal at $\mathbf{p_i}$

Considering the existence of N different surface models in the memory, Grimson and Lozano-Pérez aimed to find the best possible model in the data base that could explain the set of measurements $\mathbf{s_i}$. In order to solve this matching problem, that has complexity N^m, Grimson and Lozano-Pérez used an algorithm based on two main steps:

1. **Creation of feasible interpretations:** search in which most of the false matches are eliminated using consistency measurements such as the distance between corresponding pairs of points. In this step all the feasible interpretations of the image are generated by using geometric considerations about the shape and size of the models in the data base.
2. **Model-based test of the feasible interpretations:** the feasible interpretations created previously are tested using the surface equations obtained from the object models. An interpretation is legal if it is possible to find the rigid transformation (rotation $\mathbf{R}$ and translation $\mathbf{T}$) that places the measured points on the object surfaces.

Grimson and Lozano-Pérez calculated the rotation $\mathbf{R}$ by considering two linearly independent normals in the model, $\mathbf{m_i}$ and $\mathbf{m_j}$, corresponding to two measured normals $\mathbf{n_i}$ and $\mathbf{n_j}$ respectively. In this case $\mathbf{R}$ corresponds to a rotation of angle Θ around the vector $\mathbf{r}$, where Θ and $\mathbf{r}$ are given by:

$$\mathbf{r} = (\mathbf{m_i} - \mathbf{n_i}) \times (\mathbf{m_j} - \mathbf{n_j}) \tag{2.3.2}$$

$$\cos\Theta = 1 - \frac{(1 - (\mathbf{n_j} \cdot \mathbf{m_i})}{(1 - (\mathbf{r} \cdot \mathbf{n_i})(\mathbf{r} \cdot \mathbf{m_i}))} \tag{2.3.3}$$

$$\sin\Theta = \frac{((\mathbf{r} \times \mathbf{n_i}) \cdot \mathbf{m_i})}{(1 - (\mathbf{r} \cdot \mathbf{n_i})(\mathbf{r} \cdot \mathbf{m_i}))} \tag{2.3.4}$$

The translation $\mathbf{T}$ was obtained considering 3 linearly independent normals $(\mathbf{m_i}, \mathbf{m_j}, \mathbf{m_k})$ corresponding to 3 measurements $(\mathbf{s_i}, \mathbf{s_j}, \mathbf{s_k})$, which produced the following equations:

$$\mathbf{Rm_i} \cdot \mathbf{T} = \mathbf{Rm_i} \cdot \mathbf{p_i} - d_i \tag{2.3.5}$$

$$\mathbf{Rm_j \cdot T} = \mathbf{Rm_j} \cdot \mathbf{p_j} - d_j \quad (2.3.6)$$

$$\mathbf{Rm_k \cdot T} = \mathbf{Rm_k} \cdot \mathbf{p_k} - d_k \quad (2.3.7)$$

where:

- d_i is the distance between the face $\mathbf{f_i}$ and the origin. Notice that $\mathbf{f_i}$ is defined as the set of points $\mathbf{v}$ such as $\mathbf{v} \cdot \mathbf{m_i} = d_i$.

2.3.3 Faugeras and Herbert

One of the first pieces of research concerned with the resolution of the alignment problem was done by [Faugeras & Herbert 86] that built a system for recognition and pose estimation of objects using surface based geometric descriptions of these objects.

In their work Faugeras and Hebert described objects as sets of surfaces $\{\mathbf{P(U_i)}\}_{\mathbf{i=1,n}}$ where:

- $\mathbf{P(U_i)}$ represents an object surface (plane or quadric) and
- $\mathbf{U_i}$ is a vector of parameters that defines the surface $\mathbf{i}$

Therefore the recognition of an object modelled as $\{\mathbf{P(U_i)}\}_{\mathbf{i=1,n}}$ corresponded to the matching between $\{\mathbf{P(U_i)}\}_{\mathbf{i=1,n}}$ and a surface based description of the image being investigated: $\{\mathbf{P'(U'_i)}\}_{\mathbf{i=1,n'}}$.

The best possible model for matching the data was found by a search process whose objective was to minimise *g(m)*:

$$g(m) = min_{\mathbf{T(.)}} \quad \sum \|\mathbf{U_i} - \mathbf{T(U'_j)}\| \quad (2.3.8)$$

where:

- $\mathbf{U_i}$ is a vector of parameters corresponding to the surface $\mathbf{i}$ in the model,
- $\mathbf{T(.)}$ is a operator representing the rigid transformations that aligns the two sets of surfaces and

- $\mathbf{T}(\mathbf{U}_\mathbf{j}^{'})$ is a vector of parameters corresponding to surface **j** in the image after their alignment with the model.
- m is an integer index that indicates a given correspondence between the data and the model surfaces.

Clearly, the resolution of the registration problem is implicit in the search for the best matching. Faugeras and Hebert resolved this problem only for the case of plane surfaces that were described in their work using two parameters:

- $\mathbf{n_i}$: the ith plane normal.
- d_i: the distance between the ith plane and the origin.

For a given set of correspondences between an object model and a data set the best transformation $\mathbf{T(.)}$ (rotation $\mathbf{R}$ and translation $\mathbf{T}$) corresponding to this set of correspondences can be calculated through the minimisation of:

$$\sum_i \|\mathbf{n_i} - \mathbf{R}\mathbf{n_i}^{'}\|^2 + W_{norm}|d_i - d'_i - \mathbf{R}\mathbf{n_i} \cdot \mathbf{T}|^2 \tag{2.3.9}$$

where: W_{norm} is a weight value used to normalise the cost function.

Faugeras and Herbert minimised first the part of the Equation (2.3.9) corresponding to the rotation $\mathbf{R}$ using a quaternion representation of $\mathbf{R}$:

1.

$$\mathbf{n_i} - \mathbf{n_i}^{'} = \begin{bmatrix} v_x^i \\ v_y^i \\ v_z^i \end{bmatrix} \tag{2.3.10}$$

$$\mathbf{n_i} + \mathbf{n_i}^{'} = \begin{bmatrix} u_x^i \\ u_y^i \\ u_z^i \end{bmatrix} \tag{2.3.11}$$

$$A_i = \begin{bmatrix} 0 & v_x^i & v_y^i & v_z^i \\ v_x^i & 0 & u_z^i & -u_y^i \\ v_y^i & -u_z^i & 0 & u_x^i \\ v_z^i & u_y^i & -u_x^i & 0 \end{bmatrix} \tag{2.3.12}$$

2.

$$B_c = \sum_i A_i^T A_i \tag{2.3.13}$$

3.

$$\mathbf{R} = Q_v \tag{2.3.14}$$

where: Q_v is the rotation matrix associated with the eigenvector of B_c corresponding to the smallest eigenvalue of B_c and A_i^T is the transpose of A_i.

After the calculation of **R** the translation **T** was obtained applying a least square algorithm to minimise:

$$\sum_i \|d_i - d'_i - \mathbf{RT}\|^2 \tag{2.3.15}$$

[Marshall 89] implemented some improvements to Faugera's technique to improve its accuracy and efficiency.

Besides Faugeras and Herbert's work, another very important algorithm using quaternions to represent rotations is due to Horn [Horn & Harrys 87]. Horn developed a methodology to minimise (2.3.9) in which the best rotation was calculated by finding the eigenvector corresponding to the maximum eigenvalue of a 4×4 matrix.

2.3.4 Besl and Mackay

[Besl & McKay 92] described a general purpose and representation independent method for the accurate and efficient registration of 3-D shapes. Their algorithm, that is called **ICP** (Iterative Closest Point), is able to cope with a very large set of representations of geometric data:

- sets of points,
- line segments (polylines),
- implicit curves,
- parametric curves,
- sets of triangles and
- parametric surfaces.

In order to describe the algorithm we will first define the operators **C(P,X)** and **Q(P,Y)** where:

- **P** is a set of data points $\{\mathbf{p_i}\}_{\mathbf{i=1,n}}$
- **X** is a model shape
- **Y** is a set of model points $\{\mathbf{x_i}\}_{\mathbf{i=1,n}}$

C(P,X) returns the set of points $\mathbf{Y} = \{\mathbf{x_i}\}_{\mathbf{i=1,n}}$ such that $\mathbf{x_i}$ minimises:

$$d(\mathbf{x}) \quad = \quad \|\mathbf{x} - \mathbf{p_i}\|_{\mathbf{x} \in \mathbf{X}} \tag{2.3.16}$$

Given the corresponding point set **Y**, the least square registration between **P** and **Y** is calculated by the operator **Q(P,Y)** using the technique proposed by Horn [Horn & Harrys 87]:

$$(\mathbf{q}, d) = \mathbf{Q}(\mathbf{P}, \mathbf{Y}) \tag{2.3.17}$$

where:

- **q** is a vector containing the rotation and translation of the transformation that aligns **P** and **Y** and
- d is the value of the least square error associated with the registration between **P** and **Y**.

Having defined these two operators, the algorithm can be described as:

1. the set of points **P** and the model shape **X** are given
2. Start iteration by setting $\mathbf{P_0} = \mathbf{P}$, $\mathbf{q_0} = [1, 0, 0, 0, 0, 0, 0]^t$ and $k = 0$
3. Repeat the procedure below till convergence:
   (a) Compute $\mathbf{Y_k} = \mathbf{C}(\mathbf{P_k}, \mathbf{X})$
   (b) Compute $(\mathbf{q_k}, d_k) = \mathbf{Q}(\mathbf{P_0}, \mathbf{Y_k})$
   (c) Let $\mathbf{P_k} = \mathbf{q_k}(\mathbf{P_0})$

(d) Finish if $d_k - d_{k-1} < \epsilon_{conv}$

where: ϵ_{conv} is a small real number used to define the convergence of the algorithm.

Besl and Mackay also proposed an accelerated algorithm in which the successive estimates of $\mathbf{q_k}$ are observed and used in order to find a better direction and magnitude of change for the solution vector $\mathbf{q_k}$. The main idea is to detect when $\mathbf{q_k}$ always changes in the same direction and to use the value of d_k to find an approximation of the optimal amount of change in this direction.

It is possible to prove that the ICP algorithm always converges monotonically to a local minimum but one cannot guarantee that it will always converge to the desired global minimum. Besl and Mackay suggested running the algorithm for many initial values of $\mathbf{q_0}$ and choosing the best result. But given a set of initial values one can always come up with a set of data points for which the algorithm will not converge to the global minimum.

A similar algorithm to the ICP that uses K-D trees to improve the efficiency in the determination of the closest point was proposed by [Zhang 93].

2.3.5 Chen and Medioni

Chen and Medioni's algorithm [Chen & Medioni 92] for registration assumes the use of range data as input and is based on the minimisation of a cost function.

As Chen and Medioni said in their paper:

> ...two views of a surface are said to be in registration when any pair of points **(j,v)** from the two views representing the same surface point can be related to each other by one rigid spatial transformation **M**,

such that:

$$\forall \ \mathbf{j} \in \mathbf{J}, \ \exists \ \mathbf{v} \in \mathbf{V} \ \mid \ \|\mathbf{Mj} - \mathbf{v}\| = 0 \tag{2.3.18}$$

where:

- **J** and **V** are two views of the same surface,
- **j** and **v** are points in the views **J** and **V** and
- **M** is a 4×4 matrix representing a rigid transformation.

In practice, the problem of registering the two views **J** and **V** can be expressed as solving **M** to minimise the following measure:

$$D(\mathbf{J}, \mathbf{V}) = \sum \|\mathbf{M}\mathbf{j} - f(\mathbf{j})\| \tag{2.3.19}$$

where:

- **J**, **V**, **j**, and **M** are the same as in Equation 2.3.18.
- $f(\mathbf{j})$ is a correspondence mapping function that associates each point **j** in view **J** with its corresponding point in view **V**.

The difficulty involved in the registration problem comes from the fact that the minimisation process described above is highly non-linear when $f(\mathbf{j})$ is unknown. Also, there is no guarantee that a global minimum can be reached by an iterative procedure.

In order to solve the problem of convergence to local minimum, Chen and Medioni suggested an algorithm in which an initial approximation (M_0) of the transformation aligning the views being registered (**M**) is known, and at each iteration a better approximation of **M** is calculated.

Their algorithm can be described as:

1. Choose a set of control points $\mathbf{j_i}$ belonging to the surface view **J** and calculate the normals $\mathbf{n_i}$ at these points. Make $\mathbf{M^0} = M_0$.
2. At each iteration *k= 1,2,3, ...* repeat the following procedures until convergence:
   (a) For each surface *i*:

i. Apply $\mathbf{M^{k-1}}$ to the control points $\mathbf{j_i}$ and normals $\mathbf{n_i}$ obtaining $\mathbf{j'_i}$ and $\mathbf{n'_i}$ respectively.

ii. Find the intersection $\mathbf{v_i^k}$ of the surface $\mathbf{V}$ and the normal line defined by $\mathbf{j'_i}$ and $\mathbf{n'_i}$.

iii. Compute $\mathbf{S_i^k}$ the tangent plane of $\mathbf{V}$ at $\mathbf{v_i^k}$.

(b) Find $\mathbf{M}$ that minimises:

$$e_k = \sum_i^N \|\mathbf{M j_i} - \mathbf{v'_j}\| \tag{2.3.20}$$

$$\mathbf{v'_j} = \mathbf{v} \mid \mathbf{v} \; minimises \; \|\mathbf{M j_i} - \mathbf{v}\|_{\mathbf{v} \in \mathbf{S_j^k}} \tag{2.3.21}$$

(c) $\mathbf{M^{k+1}} = \mathbf{M M^k}$

(d) stop when $\frac{\|e_k - e_{k-1}\|}{\mathbf{N}} \leq \epsilon_{conv}$,
where: ϵ_{conv} is a real number

One of the main differences between Chen and Medioni's algorithm and others, such as Potmesil's [Potmesil 83], is that their algorithm try to minimise a distance between one point in one surface view and one surface (a plane) in the other surface view, while in other research what was minimised was the distance between two points belonging to different surfaces. The advantage of Chen and Medioni's technique is that by not finding incorrect paired points in the intermediate stage of the algorithm, they obtain a fast convergence.

2.3.6 Ponce

In a very original work [Ponce *et al.* 92] proposed an interesting technique for calculating the pose of objects in images, as well as to recognise them.

[Ponce *et al.* 92] used CAD modelling techniques to create surface based descriptions of the objects that were given by:

$$\mathbf{x}(s,t) = \frac{\sum_{i,j} s^i \cdot t^j \cdot \mathbf{x}_{i,j}}{\sum_{i,j} s^i \cdot t^j \cdot w_{i,j}} \tag{2.3.22}$$

where:

- $w_{i,j}$ are real numbers,

- $\mathbf{x}_{i,j}$ are 3-D vectors and
- s and t are variables used to parameterise the surfaces of the object.

The main idea in their work is the derivation of equations like:

$$\mathbf{f}_1(s, t, \mathbf{O}, \mathbf{Par}) = 0 \tag{2.3.23}$$

$$\mathbf{f}_2(s, t, \mathbf{O}, \mathbf{Par}) = 0 \tag{2.3.24}$$

$$\mathbf{f}_3(s, t, \mathbf{O}, \mathbf{Par}) = 0 \tag{2.3.25}$$

where:

- s, t are the same as in Equation (2.3.22),
- $\mathbf{O}$ is a vector of observables extracted from the object image and
- $\mathbf{Par}$ is a vector of parameters corresponding to the pose of the objects.
- $\mathbf{f}_i$ represents a relation between the variables s and t, the vector of observables $\mathbf{O}$ and the vector of parameters $\mathbf{Par}$.

Through the elimination of s and t in Equations (2.3.23) to (2.3.25) it is possible to obtain one unique equation relating the vector of observables $\mathbf{O}$ and the vector of parameters $\mathbf{Par}$ corresponding to the object position:

$$\mathbf{F}(\mathbf{O}, \mathbf{Par}) = 0 \tag{2.3.26}$$

This means, that the vector of parameters $\mathbf{Par}$ can be determined by using the vector of observables $\mathbf{O}$ and solving Equation 2.3.26.

If we now consider a set $\{\mathbf{O_i}\}_{i=1,n}$ of parameter vectors associated with one image, the best pose estimate corresponds to the vector $\mathbf{Par}$ that minimises the expression:

$$g(\mathbf{Par}) = \sum_{i=1}^{n} \mathbf{F}(\mathbf{O_i}, \mathbf{Par})^2 \tag{2.3.27}$$

Thus, the estimation of the object pose in the image can be done through the algebraic manipulation of Equations (2.3.23) to (2.3.25) in order to obtain Equation (2.3.27) that is then minimised in $\mathbf{Par}$.

[Ponce *et al.* 92] pointed out the possibility of using many kinds of input data and thus many kinds of vectors of observables **O**:

- range data,
- orientation and gaussian curvature,
- intensity image,
- intensity image and gradient intensity image and
- image contour

As an example of Ponce's technique let us consider the case of range data in which the vector of observables corresponds to **O** = *(x, y, z)*. Where x and y are the orthographic projection on the image plane of a point p and z is its depth.

In this case, if we considered the image reference frame defined by $(\alpha, \beta, \gamma, x_0, y_0, z_0)$, where (α, β, γ) are related to the orientation of the image frame axes and (x_0, y_0, z_0) with its origin, by applying the general principle explained before we can get a single equation relating the vector of observables **O** = *(x, y, z)* and the vector of viewing parameters **Par** $= (\alpha, \beta, \gamma, x_0, y_0, z_0)$:

$$\mathbf{F}((x, y, z), (\alpha, \beta, \gamma, x_0, y_0, z_0)) \tag{2.3.28}$$

If we now consider different vectors of observable $\mathbf{O_i}$, the image reference frame can be determined by minimising in $(\alpha, \beta, \gamma, x_0, y_0, z_0)$ the equation:

$$g(\alpha, \beta, \gamma, x_0, y_0, z_0) = \sum_{i=1}^{n} \mathbf{F}(\mathbf{O_i}, (\alpha, \beta, \gamma, x_0, y_0, z_0))^2 \tag{2.3.29}$$

2.3.7 Stein

[Stein 92] and [Stein 91] proposed the use of indexing together with search in an interpretation tree, as advocated in [Grimson & Lozano-Perez 84], for the recognition of objects in binary intensity images, range images and grey level images. Stein used the indexing for generating initial hypotheses of matching that are tested by search in an

interpretation tree. Through the indexing of local properties Stein managed to make his method insensitive to occlusion as well as reduced the computational complexity of the search for the correct matches between model and data.

In Stein's method, the local topological information or the local geometrical properties are coded and stored in a memory address corresponding to their code. Model features whose properties have the same value are stored at the same address in a hash table (model data base), which is built in a pre-processing stage. The recognition process is divided in two main steps:

1. Apply the coding procedure to the image and generate initial hypotheses for search in the interpretation tree. The initial hypotheses are generated by considering the model and data features with the same code in the hash table (model data base).

2. Use of traditional search techniques for the identification of models present in the data and calculation of the rigid transformation aligning these models to the data.

In the case of binary images, the indexing is done by finding the outline of the binary images using edge detection and approximating this outline by a polyline. Indexing codes are generated by encoding the angle between successive lines in the polyline. In order to guarantee robustness the codes are generated by considering different numbers of line segments and different starting points in the polylines. Polylines with different degrees of approximation to the original outline are also used. Results are presented in binary images with high degree of occlusion.

With range images, Stein used a new local geometrical feature called the *splash* for indexing. Splashes correspond to a set of 3D polylines that encodes the angles between the normals belonging to a local patch on the surface of the object. As before, different degrees of approximation and starting points are used to make the method more robust. Results are presented with range images of human heads.

Intensity images are encoded using invariant properties of the edges in the image. Initial results are presented for some simple real images.

2.3.8 Guéziec and Ayache

[Gueziec & Ayache 94] presented a new approach to matching 3D curves that has computational complexity independent of the number of models. Their approach is an enhancement of [Kishon *et al.* 90] with a more accurate estimation of pose and a smaller computational complexity.

The main objective of Guéziec and Ayache was the recognition of 3D objects using the following strategy:

- **Characteristic curve extraction:** Extraction of curves describing highly structured descriptions from 3D images such as surface borders, borders of holes, junctions, ridges, parabolic lines, umbilical points, etc. [Ayache *et al.* 89].
- **Matching:** Matching of characteristic curves for the final recognition.

The matching of the characteristic curves was implemented by Guéziec and Ayache by:

1. Approximation of the characteristic curves using nonuniform B-splines that were used to estimate torsion and curvature. Guéziec and Ayache used an elaborate spline fitting that minimised not only the square distance between the curve approximation and the data points, but also other terms corresponding to curvature and normals variation. Their objective was to obtain a more accurate estimation of torsion and curvature in the presence of noise that allowed estimation of the Frenet frames associated with the characteristic curves.
2. Matching of the model and data curves using the Frenet frames. Three different approaches were discussed for the matching: prediction verification, Hough techniques and model indexing. The most efficient one was the latter. Guéziec and Ayache used the geometric hashing method suggested by [Kishon *et al.* 90] to implement the model indexing. The curvature of the B-spline approximation of the characteristic curves was used as an invariant to index the models.

Results of the matching of C-T images of a vertebra and a skull are shown. Typical

times for the matching stage are of the order of a few seconds on a DEC 9500.

2.4 Modelling solid objects: nominal shape

Because of its importance in many different areas such as computer vision, CAD/CAM and automated inspection, the modelling of solid objects has been the aim of a great deal of research. In this section we review the most important representation schemes used for solid modelling.

A representation scheme can be defined as a mapping relating a mathematical model of a solid to a set of symbolic structures [Juster 92]. In order to be able to calculate some geometric properties of the objects they describe, representation schemes must have some formal properties (see [Requicha 80] and [Zeid 91]):

- **Domain**: the class of objects that the scheme can represent.
- **Validity**: determined by the range of the representation scheme, that is, the set of representations which are valid (*sensible*) euclidian objects.
- **Completeness**: related to the ability of the scheme to support analysis; a complete scheme should provide models with enough information to perform any necessary geometric calculation.
- **Uniqueness**: this property is useful for determining object equality. A given representational scheme presents uniqueness when there is a one-to-one mapping between the representations (models) and the euclidian objects represented.

There are other desirable properties of representation schemes as conciseness, facility to build and efficiency. These properties are considered informal because they cannot be easily formalised.

In the next sections we review the main characteristics of the most important representation schemes used in solid modelling:

- wireframes,

- cell decomposition,
- sweep representations,
- constructive solid geometry (CSG) and
- boundary representations (B-rep).

A more extensive discussion about these representation schemes can be found in [Besl 88b], [Requicha 80], [Ballard & Brown 82] and [Zeid 91], from which this review draws heavily.

It is important to observe that this section aims only to review some of the essential characteristics of the most commonly used representation schemes. The discussion about the use of these schemes for inspection purposes is presented in Chapter 3.

2.4.1 Wireframe models

Wireframe models use edges as the primary entity for representing objects. The word wireframe comes from the fact that one can imagine a wire that is bent to follow the object edges to construct the model. Wireframe models consist entirely of points, lines, arcs, circles and curves. Since their first appearance in the early 1960s, for the representation of 2D objects, the wireframe modelling technique became very popular and is the most commonly used technique used CAD/CAM systems nowadays [Zeid 91].

The main advantage of wireframe models is their simplicity that makes it easy to construct, store and use them. Among their many limitations, the biggest problem is their ambiguity, even for the restricted domain of polyhedral objects [Requicha 80]. Wireframe models also lack information for determining the object profile; and models of complex objects may be difficult to understand.

2.4.2 Cell decomposition and spatial occupancy enumeration

Cell decomposition models represent volumes as three dimensional arrays of cells that may be marked as filled or not. Spatial occupancy representation corresponds to a

particular case in which the cells are cubes of fixed size lying in a regular grid. In both cases two distinctive cells of the model cannot share the same volume in the 3D space.

Cell decomposition models are often difficult to construct and their conversion to other representations is not an easy task. Two of the most common techniques for the recursive decomposition of volumes into cells are the oct-tree [Jackins & Tanimoto 80] and the kd-tree [Bentley 75].

The calculation of mass properties and the representation of inhomogeneous objects are some of the common applications of cell decomposition models. Clearly, the main disadvantage of this representation scheme is the excessive verbosity.

2.4.3 Sweep representations

The movement of a 2D set along a curve in the 3D space describes a volume. General sweep representations use this fact to describe volumes in terms of a moving set, a trajectory and a sweeping transformation (which is often used for scaling or rotating the moving set). General sweep representations are also known as generalised cylinders or cones [Zeid 91].

General sweep schemes are intuitively very appealing and naturally describe many biological and manufactured objects, as well as some manufacturing processes such as material removal and dynamic interference. The main problem related to the use of sweep representations is the absence of a formal theory of sweeping. For example, general validity conditions for sweep representations are unknown [Requicha 80].

2.4.4 Constructive Solid Geometry (CSG)

CSG models are based on the topological notion that a physical object can be understood as a set of primitives combined in a certain order following a set of rules [Zeid 91].

As combination rules CSG schemes use Boolean operations (*e.g.* union, difference and intersection).

There are two main types of CSG schemes. The most popular one uses bounded

solids (*e.g.* cubes, cylinders and cones) as primitives. In this case each primitive is parameterised to define its size and pose. The second kind of CSG scheme uses unbounded half-spaces as primitives. The schemes based on the use of bounded solids are usually more concise because half-spaces are lower-level primitives.

CSG representations are also succinct, easy to create, to manipulate and to store. CSG schemes provide an easy way to understand material removal processes and interference checking between objects.

2.4.5 Boundary representations (B-reps)

B-rep and CSG are the two most popular and widely used representation schemes for modelling solid objects [Zeid 91].

B-reps model solid objects by describing the set of faces that bind these objects. The faces in a B-rep are bound by their edges, and the edges in a surface are bound by their vertices. Thus, a boundary model of a solid consists of sets of faces, edges and vertices that are linked in such a way as to ensure topological consistency.

A very influential system for using face-based representations for polyhedral objects was the winged edge representation [Baumgart 72]. This system included an editor for creating complex objects using the Euler theorem as construction rule.

Boundary representation schemes in general are very verbose and difficult to build, though they have the advantage of representing explicitly some useful information (*e.g.*, faces and edges) that is essential in many applications (*e.g.* inspection and object manipulation).

2.5 Modelling solid objects: tolerancing

Because manufacturing methods are incapable of producing parts with perfect shapes and sizes, an essential point in the design of mechanical parts is the specification of the parts tolerances. The complete design of a mechanical part implies not only the definition of a nominal or ideal part, but also the definition of the acceptable variance from the nominal part.

In the current industrial practices, the dimensions and tolerances of mechanical parts are represented by engineering drawings that represent the ideal shape and the acceptable variances according to international [R1101 82], national [Y14.5M 82], [BS308 72a] and [BS308 72b] or even company standards.

The dimensioning and tolerancing standards describe two types of tolerances [Zeid 91]:

- the traditional *plus/minus* dimensioning tolerances that control the part's size and
- the geometric tolerances.

Conventional tolerancing control only the variability of linear dimensions that describe the location, size and angle. Therefore conventional tolerancing schemes are incapable of controlling all aspects of the shape of a part. Other shortcomings of the traditional tolerance methods are that they do not use the datum concept (reference used to locate features during the manufacturing and inspection), and that they are not suitable to control locations in 2D and 3D models. To overcome all these drawbacks the geometric tolerancing was created.

A fundamental element in geometric tolerancing is the concept of datum which is a reference for locating features during manufacturing and inspection. Datum planes correspond to abstractions representing surfaces of machine tables, fixtures, inspection devices and gages. Datum features are the actual features in the part used to define the datum planes.

Geometric tolerances are divided in four categories [BS308 72b]:

- **form tolerances**: verify the deviation of shape.
- **attitude tolerances**: check the relative orientation of features.
- **location tolerances**: restrain the relative position of features.
- **runout tolerances**: control the movements of the axis of rotation of rotational parts.

A part is in tolerance if all of its tolerance variables are within the acceptable variations defined by the tolerance specification [Juster 92].

Several different approaches have been proposed for combining the dimensioning and tolerancing standards with solid modelling to produce models of variational classes [Requicha 77], *i.e.* the set of all physical components that satisfy a given set of tolerance specifications. Unfortunately, mainly due to the informal definitions given in the current standards, none of these approaches have been completely successful.

Besides creating a mathematical model of the semantics of the current dimensioning and tolerancing standards, another important issue related to tolerancing is the development of methods for representing the syntax of the tolerancing standards.

Section 2.5.1 to 2.5.3 review the research done concerning the semantics of tolerancing and Section 2.5.4 reviews the different methods concerned with the syntax of the tolerancing.

More extensive reviews about these topics can be found in [Requicha 80], [Zeid 91] and [Juster 92] from which the next sections draw heavily.

Section 2.5.3 and Section 2.5.4 aim only the review of the main research done in the area. The particular approach we adopted for the definition of a semantics and a syntax of tolerancing for inspection purposes is discussed in Chapter 3.

2.5.1 Tolerancing semantics: parameterisation

Most of the theory developed to model variational classes is based on the parameterisation of the object being modelled. In this case, shape and size are controlled through the variation of the model parameters. The parameterisation is called *direct* when the vector of model parameters $\mathbf{m}$ can be calculated as an explicit function ($g(.)$) of the vector of toleranced dimensions $\mathbf{t}$. The parameterisation is called indirect or inverse when only the inverse of the function $g(.)$ ($h(.) = g^{-1}(.)$) is known explicitly.

Figure 2.1 illustrates the direct and the indirect parameterisation in the case of a rectangular hole whose toleranced dimensions are $\mathbf{l}$ and $\mathbf{w}$ and the model parameters are the coordinates of the vertices of the hole.

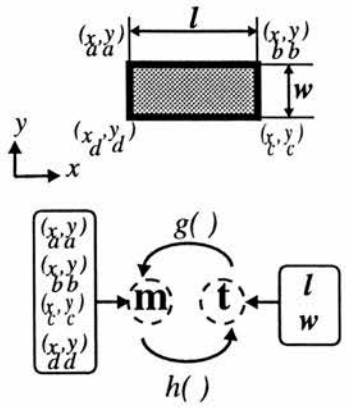


Figure 2.1: Direct and indirect parameterisation in rectangular hole.

With both parameterisations, the variational class of a given model is defined by the set of perfectly formed objects produced when the vector of toleranced dimensions $\mathbf{t}$ assumes values in a region of the vector space spanned by $\mathbf{t}$.

Determining the invertibility of $h(.)$ it is not a trivial task and has been studied by several authors.

[Hillyard 78], [Hillyard & Braid 78] and [Hillyard & Braid 82] proposed a theory that tries to explain the how the combination of dimensions and views in an engineering drawing specifies the shape of mechanical components. Hillyard modelled objects as engineering frame structures whose members and joints are the edges of the object. In Hillyard's approach adding dimensions to the object is equivalent to adding stiffeners to the frame structure. Variational classes are modelled as variations around the nominal shape using flexibility matrices that relate small variations in the model parameters to small variations in the toleranced dimensions.

[Light 80] generalised the work of Hillyard and constructed a system called DIMENSION that allowed the on-line modification of the dimensions of the object during the design phase. Light worked with 2D objects that he modelled as a set of points connected by lines. Light's techniques were extend to 3D problems in [Lin *et al.* 88].

[Requicha 77] proposed a data structure for representing dimensions using forward chaining. In the forward chaining, each feature is defined in relation to its predecessor feature forming a chain. This data structure was used by [Gopin & Gossard 79] to develop a system that allowed control of the distances between parallel lines. In this

system, data was represented as two directed graphs that become trees when the part is properly constrained. [Fitzgerald 81] and [Juster 88] also developed systems using methods based on the dimension trees of Gopin and Gossard.

[Todd 86] proposed an extension of the dimension trees that allowed orientation of 2D straight-line segments at any angle. [Parden & Newell 84] developed the MEDUSA system in which it is possible to control the dimensions between pairs of circles or lines.

[Suzuki *et al.* 85] and [Kimura *et al.* 86] used first order predicate logic to define dimensions of 3D objects. A similar version for 2D objects was implemented by [Aldelfeld 86].

[Gossard *et al.* 88] implemented an extension of [Gopin & Gossard 79] work for 3D objects. Dimensions are represented as nodes in a graph that point to a B-rep of the object. When a dimensional change is made, the object graph is changed from bottom-up and a new B-rep of the object is produced. The system allows control of the parallel distance and intersection angle of planar surfaces.

[Bernstein & Priess 89] also used graphs for modelling solid objects. In their approach, however, the nodes of the graph corresponded to the faces of the object and the links between the nodes corresponded to transformation matrices that constrained the faces' relative position.

[Cardew-hall *et al.* 93] proposed a method based on [Bernstein & Priess 89] constraint propagation approach. [Cardew-hall *et al.* 93] developed an extended theory based on local feature datum systems. The scheme was also implemented and used to produce a CAD model of a turbine blade for inspection purposes.

2.5.2 Tolerancing semantics: vector space of model variables

In all the works of the previous section, the variational class models were defined by constraining the vector of toleranced dimensions $\mathbf{t}$ to a region in the vector space spanned by $\mathbf{t}$. The main difficulty with this approach was to guarantee that each of the allowable values of $\mathbf{t}$ corresponded to a valid object.

A possible way to escape this problem is to define the variational class models by

constraining the vector of model variables **m** to a region in the vector space spanned by the model variables.

This idea was first used by [Hoffman 82] in the case of two dimensional objects. Hoffman modelled the edges of the 2D objects using two model variables and constrained these variables using linear inequalities.

An extension of Hoffman's work to the three dimensional case was developed by [Turner 87]. Turner developed the GEOTOL system for tolerance analysis using a CSG modeller from IBM - the GDP. In Turner's work the models B-rep consisted of planar and cylindrical patches. Variational models were defined by constraining the variations in the pose and dimensions of patches in the B-rep. Like Hoffman, Turner used tolerance functions to impose linear or almost linear constraints to the model variables.

2.5.3 Tolerancing semantics: offset models

[Requicha 83] proposed a mathematical theory of tolerancing that tries to formalise and generalise the current practices defined in the standards. Requicha's main aim was to allow the construction of geometric solid modelling systems able to deal with tolerances.

According to Requicha an actual feature f_a is within tolerance t if the feature lies inside a tolerance zone defined by offsetting (see [Requicha 86] for a formal definition of offsetting) the corresponding nominal feature f_N by a positive distance d_p and a negative distance d_n such that:

$$t = d_p + d_n \tag{2.5.30}$$

Each feature is given tolerance values which define acceptable variations in:

- **size:** – the tolerance zone has a fixed size, but its position and its orientation are allowed to change.
- **shape:** – the tolerance zone may be translated, rotated and scaled (as long as t is kept constant).

- **orientation:** – the tolerance zone has a fixed size and orientation.
- **position:** – the tolerance zone has a fixed size, orientation and position.

Requicha's theory departs from the current standards in some cases, and its main problem is the definition of the tolerance zones as a function of the functional requirements of the part. An interesting discussion of the viability of Requicha's ideas is presented in [Juster 92].

A very similar approach to Requicha's is used by [Etesami 88].

2.5.4 Tolerancing syntax

The most significant work concerning the representation of tolerancing syntax is being done in the ISO and ANSI standards that are developing the STEP (STandard for the Exchange of Product model data) and the PDES (Product Data Exchange Specification) respectively [Juster 92]. At the moment the two standards are equivalent.

Neither the STEP nor the PDES address the problem of adding tolerance information to solid modelling systems. This problem was addressed by [Johnson 85] that considered a solid modelling system composed of a virtual solid modeller and an application interface (AIS).

[Johnson 85] also proposed an extension of the AIS for the construction of EDT (Evaluated Dimension and Tolerancing) models based on the current American standards [Y14.5M 82]. EDT models are constructed by building a model of the nominal shape (using the AIS interface standard) and then linking this solid model to DT (dimension and tolerancing) nodes that define the tolerances associated with the model of the nominal shape.

[Roy & Liu 88] also studied the problem of attaching tolerances to CSG and B-rep models. They identify two levels of feature:

- **low-level features**: such as vertices, edges and surfaces.
- **high-level features**: the combination of the lower level features.

According to them, B-rep models are better for the lower level features and CSG models are better for high-level features. This hierarchical division of features was also used by [Gao *et al.* 93] that classified the features as atomic, primitive and compound. [Gao *et al.* 93] implemented a system able to cope with some common features in the manufacturing industry as bosses, pockets, holes, slots, notches, steps and surfaces.

2.6 Conclusions

In this chapter we presented a short literature review of the significant research related to the design and implementation of automated vision inspection systems. Automated inspection, data acquisition, registration, geometric modelling and tolerancing were discussed.

The comparison between our main framework of research and the literature review presented in this chapter reveals some important facts:

- **3D Data Acquisition:** Despite the clear fragility of some techniques that are still not advanced enough for their use for inspection purposes, we already have some commercial systems able to deliver the necessary accuracy and robustness necessary for the inspection of mechanical parts in the manufacturing environment.

- **Modelling Issues:** The representation of the nominal shape of manufactured objects, as well as the tolerances associated with these nominal shapes is still a problem far from being satisfactory solved. We present our particular approach to this issue in Chapter 3.

- **Registration Techniques:** We have a considerable number of accurate, robust and well known techniques for the registration between model and data. Furthermore, the application of some of these techniques for inspection purposes (in the case of objects with enough distinctive features to guide the registration) has already been object of very careful analysis [Marshall 89]. Despite this fact, there is a lack in the analysis of the application of these registration techniques in the case of sculptured surfaces. This means that the inspection of sculptured surfaces

demands a careful analysis of the application of the optimisation techniques for inspection and of the limitations of these techniques. We shall discuss the use of these techniques for inspection purposes in Chapter 4.

- **Inspection Diagnosis**: Section 2.1 shows how the research done aiming the use of range data for inspection purposes was never concerned with the formal definition of the inspection procedures or with the numerical evaluation of the quality of the inspection procedures developed. We tackle this issue in Chapter 4. Furthermore, we give special attention to the problem of inspecting sculptured surfaces using dense range data (Chapters 5 and 6) in an extension of [Marshall 89], [Marshall *et al.* 91] and [Newman & Jain 95].

Chapter 3

Model data base

This chapter presents our approach for modelling objects for inspection purposes. In Section 3.1 we discuss the different possible representation schemes and present our choices for representing nominal shape and tolerances. Details of our particular implementation are discussed in Sections 3.2 and 3.3. Section 3.4 presents our final conclusions.

3.1 Prototype representation

The model data base is one of the most important modules in our framework of research. The model data base stores the information about the nominal shape of the parts to be inspected, as well as the tolerancing information concerning the acceptable variations in the geometry of the parts. Clearly, the main issue involved in the design of this module is a representational one: *what is the best way of representing objects for inspection?*

There is not a general and satisfactory answer to this question, but there are some important characteristics of the models that are desirable [Marr 82] [Requicha 80], [Fisher 89], [Marshall 89]:

1. to represent the shape of objects and to be able to infer from it how the shape affects the appearance of objects in different points of view,

2. to represent explicitly the necessary information for inspection,

3. to be able to cope with the range of objects pertinent to the inspection task,
4. to be unambiguous,
5. to be able to represent tolerances in the shape of the objects,
6. to be able to represent the features of the objects with different degrees of detail as required,
7. to be easy to create and to manipulate, and
8. to produce succinct representations.

From now on we shall call these characteristics as *desirable properties* of the representation schemes.

The analysis of the main characteristics of the representation schemes described in Section 2.4 shows that these schemes are either completely inadequate or not ideal for inspection purposes:

- **wireframe:** ambiguous, it does not represent surfaces which makes its use with sculptured surfaces impossible.
- **spatial occupancy enumeration and cell decomposition:** they do not explicitly represent features such as edges or surfaces and are too verbose.
- **sweep representation:** unable to cope with many objects and not very well understood.
- **CSG:** it does not portray boundary information explicitly.
- **B-reps**: verbose and difficult to build.

The dimensioning and tolerancing of mechanical parts, as described in the standards ([R1101 82], [Y14.5M 82], [BS308 72a], [BS308 72b]), relies heavily on using features such as surfaces or holes in the surface boundary. Therefore, it is essential to any representation scheme used for inspection to represent these features explicitly.

We can eliminate from start the wireframe, the spatial occupancy enumeration, the cell decomposition and the sweep representations that are clearly completely inadequate. CSG schemes are very common in manufacturings, but they have the drawbacks of not representing most of the important features explicitly. By taking into account all these facts we decided to investigate the use of B-reps for inspection purposes.

Figure 3.1 shows a numerical evaluation, built using our personal evaluation of the desirable properties enumerated previously, of the different representation schemes. This figure represents an attempt to compare the different representation schemes according to each of the desirable properties and, consequently, to justify our choice for the use of B-reps for inspection purposes.

It is important to observe that Figure 3.1 was built using very personal and subjective criteria. Therefore, the figure should only be taken as an illustration of our ideas towards the problem of representing objects for inspection.

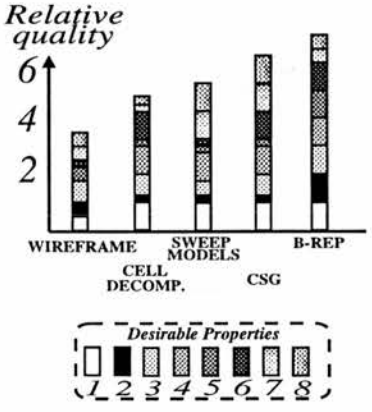


Figure 3.1: Relative quality of different representation schemes (wireframes, cell decomposition, sweep representation, CSG and B-reps) according to the desirable properties of the representation schemes.

B-reps can only represent the nominal shape of objects, and it is necessary to attach some tolerancing information to them before they can be used for inspection.

As it was shown in Section 2.4, the problem of defining a semantics of tolerancing is far away from being solved and none of the many different solutions proposed is ideal. In our work we decided to follow the current standards closely, while attempting to adapt Requicha's theory of tolerancing to them. We adopted this strategy because we wanted to use a tolerance semantics able to cope with the current standards, moreover

we also believe that Requicha's theory is the most suitable to define a semantics of tolerancing. Our preference for Requicha's theory is due to many considerations:

- **Simplicity:** the basic concepts of Requicha's theory are very simple and intuitive.
- **Syntax:** The syntax of Requicha's theory can be implemented in a way very similar to the standards.
- **Semantics:** Although the exact interpretation of Requicha's theory is not well defined in many situations, there are a few cases in which its meaning is clear and close to the standards.

It should be mentioned that Requicha's theory is an attempt to generalise the ideas present in the standards and, it comes from this fact, the similarity between Requicha's ideas and the standards in many situations. However, there are many situations described in the standards that are not covered by Requicha's ideas as, for instance, unilateral tolerance specifications.

By taking into account all these facts, we decided to use EDT models and DT nodes (see Section 2.5) to represent the syntax of tolerance. Our choice was based on two main reasons:

- DT nodes represent the tolerances as described in the ANSI and ISO standards and
- DT nodes can be easily extended to represent the *position*, *shape* and *size* tolerance zones as described by Requicha (see Section 2.4.3).

Figure 3.2 illustrates the main idea behind the construction of an EDT model that is composed of two main elements:

- a B-rep that lists all the surfaces, curves and vertices in the object.
- a list of DT nodes that defines the tolerances associated with the surfaces, curves and vertices in the B-rep.

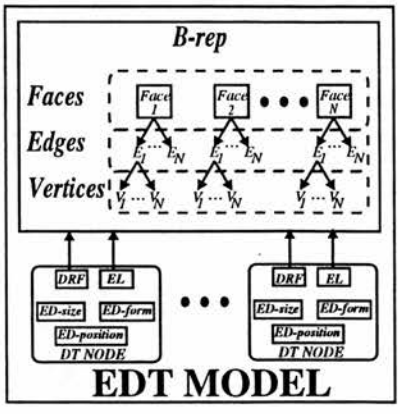


Figure 3.2: Illustration of EDT model.

The DT node is a data structure composed of the follow elements [Zeid 91] (see Figure 3.2):

- **Entity Linking node (EL)**: links the DT node to the set of features (vertices, edges or faces) in the B-rep whose tolerance is defined by the DT node.
- **Datum Reference-Frame (DRF)**: links the DT node to the set of features (vertices, edges or faces) in the B-rep that defines the reference datum used to define the tolerance.
- **ED-size, ED-position and ED-form**: define the tolerance values of the tolerance defined by the DT node.

3.2 Implementation: Boundary representation

We implemented a boundary representation scheme based on the hierarchical tree shown in Figure 3.3 in which each element in the tree inherits the attributes of its predecessors.

At the top of the tree the basic element in our representation is the feature *Feature* that has the following attributes:

- NAME: feature identification.

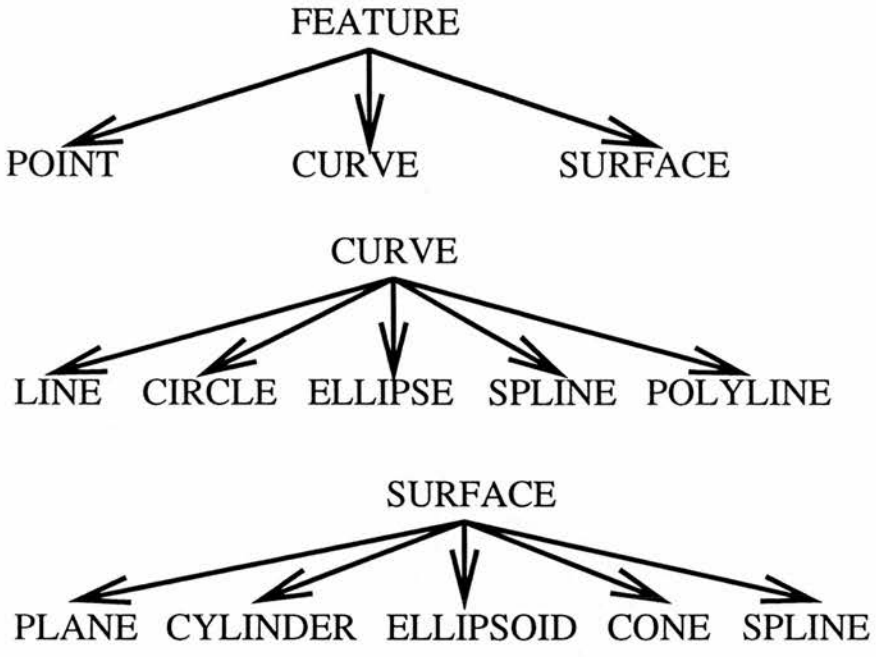


Figure 3.3: Boundary representation data structure.

- FEATURE TYPE: kind of feature - point (*Point*), curve (*Curve*) or surface (*Patch*).
- POSE: position and orientation of the feature in relation to its canonical position. See below for the definitions of the canonical positions of different *Feature* types.

A *Point* is a *Feature* corresponding to a position in the 3D space. Besides all the attributes of a *Feature* a point also has the following attributes:

- LOCATION: position of *Point* in the canonical frame (usually points are defined with null POSE).
- CURVES: list of pointers to all the *Curves* the point belongs to.
- SURFACES: list of pointers to all the *Patches* the point belongs to.

Three dimensional curves are represented by *Features* of kind *Curve*. *Curves* have the following attributes:

- CURVE TYPE: line (*Line*), circle (*Circle*), ellipses (*Ellipses*), spline curve (*Splinecur*), list of continuous curves (*Polycurve*).

- BEGIN POINT: starting *Point* of curve.
- END POINT: end *Point* of curve.
- SURFACES: list of pointers to all the *Patches* the curve belongs to.

The canonical position and attributes of *Lines*, *Circles*, *Ellipses* and *Spline-cur* can be seen in the Figure 3.4.

- Lines are defined with as parallel to the z direction and with the begin point at the origin of the canonical reference frame.
- Circles and Ellipses are defined as curves in the plane xy from the begin point to the end point in the clockwise sense.
- Spline curves are defined as non-rational uniform B-splines with the begin point at the origin of the canonical reference frame.

A *Polycurve* is a *Curve* composed by a set of other *Curves*. The only attribute of a *Polycurve* is a list of pointers to the *Curves* that compose it.

A *Patch* corresponds to a *Feature* used for representing patches, or pieces of infinite surfaces. The attributes of a *Patch* are:

- SURFACE TYPE: plane (*Plane*), cylinder (*Cylinder*), cone(*Cone*), ellipsoid (*Ellipsoid*), spline surface (*Spline-surf*).
- INCLUDED POINT: *Point* belonging to the interior of the patch.
- LOOP: list of closed curves (*Polylines*) that delimit the patch extension.

The canonical position and attributes for all the different surface types are described in the following (where **C. Pos** stands for canonical position and C.R.F. for canonical reference frame):

- **PLANE**

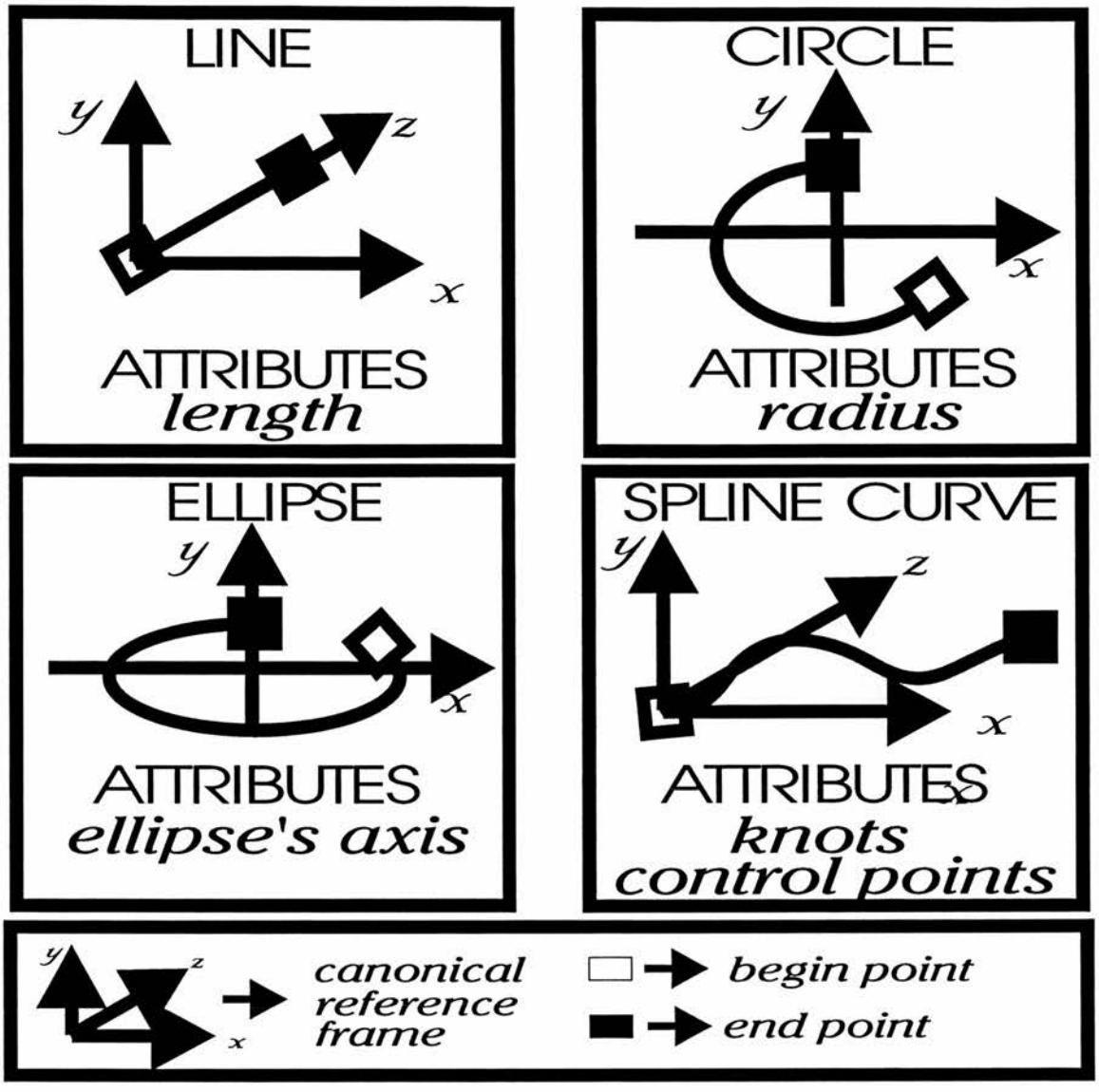


Figure 3.4: Curve attributes and canonical position.

- **C. Pos.:** any but should pass by origin of C.R.F.
- **Attributes:**
  - normal direction

- **CYLINDER**
  - **C. Pos.:** axis on X axis of C.R.F.
  - **Attributes:**
    - starting and ending point of cylinder's axis on X axis of C.R.F.
    - ellipse curve of cylinder's base

- **ELLIPSOID**
  - **C. Pos.:** center at C.R.F. origin
  - **Attributes:**
    * size of main axis of ellipsoid
    * starting and ending points of ellipsoid patch (parameterised in spheric coordinates)
- **CONE**
  - **C. Pos.:** axis on X axis of C.R.F
  - **Attributes:**
    * position of cone apex on X axis of C.R.F.
    * starting and ending points of cone's axis
    * circle curve secting cone on starting point
    * circle curve secting cone on ending point
- **SPLINE SURFACE** (non-rational uniform B-spline)
  - **C. Pos.:** any (Monge projection)
  - **Attributes:**
    * number of knots
    * knots
    * control points

Figure 3.5 illustrates the canonical position of the surfaces in the C.R.F. (canonical reference frame).

The complete boundary of the object being modelled is built by storing all the vertices of the object in *Point* structures, all the edges in *Curve* structures and all the surfaces in *Patch* structures. The final B-rep is stored in a data structure called *B-rep* whose attributes are:

- NAME: identify the B-rep.

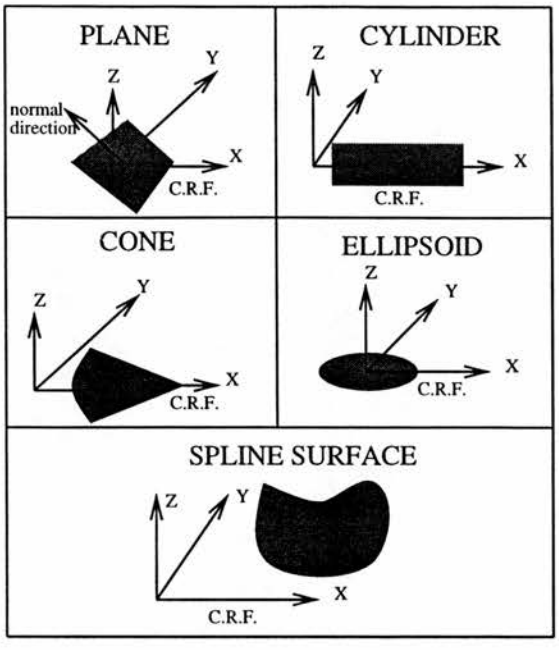


Figure 3.5: Surfaces canonical position (C.Pos.) in the canonical reference frame (C.R.F.).

- POINTS: list of pointers to all the *Points* in the B-rep.
- CURVES: list of pointer to all the *Curves* in the B-rep.
- SURFACES: list of pointer to all the *Patches* in the B-rep.
- ADJACENCY: data structure with information about the adjacency between the patches in the B-rep. This data structure enumerates all the pairs of adjacent patches in the B-rep (see Appendix C for an example of how the adjacency information is stored).

An example of a complete B-rep model can be seen in Appendix C. This appendix contains the list of all the vertices, edges and surfaces of the widget shown in Figure 4.44.

3.3 Implementation: EDT models

A complete EDT model is composed of a B-rep and a set of DT (Dimensioning and Tolerancing) nodes pointing to the features in the B-rep. We implemented this structure using a data structure whose attributes are:

- NAME: to identify the EDT-model.
- B-REP: a pointer to a B-rep data structure (as described previously).
- DT-NODES: a list of pointers to DT nodes that associate tolerances with the B-rep.

Each DT-node was implemented as illustrated in Figure 3.2 and is composed of the following attributes:

- NAME: identification of DT node.
- LIST OF FEATURES: list of pointers to the features the DT node refers to (EL in Figure 3.2).
- DRF: data structure that defines the *datum reference frame* of the DT node.
- SIZE: data structure ED-size that defines the *size* tolerance associated with the DT nodes.
- FORM: data structure ED-form that defines the *form* tolerance of the DT node.
- POSITION : data structure ED-pos that defines the *position* tolerance of the DT NODE.

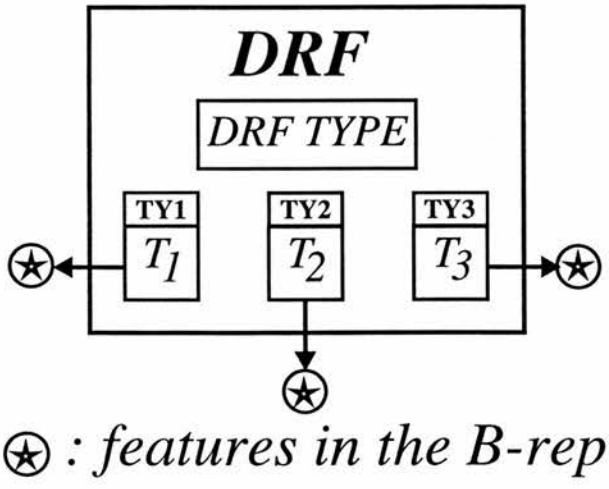


Figure 3.6: The DRF structure - Datum Reference Frame.

The DRF structure (see Figure 3.6), that defines the *datum reference frame*, presents the following attributes:

- DRF TYPE: defines the kind of datum used: NULL, THREE-FACES, ONE-FACE and ONE-LINE.
- DATUMS T1, T2 and T3: lists of pointers to B-rep features defining the datums T1, T2, T3.
- DATUMS TYPE TY1, TY2 and TY3: they define the type of the datums T1, T2 and T3 respectively. The possible datum types are: NULL, FACE, TWO-EDGES, EDGE-VERTEX, THREE-VERTICES and TWO-VERTICES.

In the DRF node, the DRF TYPE defines the kind of datum used by the tolerance:

- THREE-FACES means the definition of a complete 3D datum reference frame using three orthogonal planes. For instance, the THREE-FACES datum can be used to define the position of a vertex in the 3D model of the nominal shape.
- ONE-FACE means a definition of a 2D datum reference frame using an unique plane. This kind of datum can be used to define the position of the center of a hole in a planar surface, for example.
- ONE-LINE means the definition of a 1D datum reference frame using an unique line. This kind of datum can be used to define the length of a segment of line, for example.

The DATUM TYPE attribute characterises the feature(s) that defines each datum in the DRF NODE. For instance, we need three planes in a THREE-FACES DRF node. Each of these faces can be defined by many different features: for example, a plane (FACE), a point and an edge (EDGE-VERTEX), three points (THREE-VERTICES), two edges (TWO-EDGES) and other combinations of the basic types of datums.

The ED-form, ED-size, and ED-position data structures define the *form*, *size* and *position* tolerances in the DT node respectively. Figure 3.7 illustrates the ED-form structure, the ED-size and the ED-position structures are basically the same. These three structures present the following attributes:

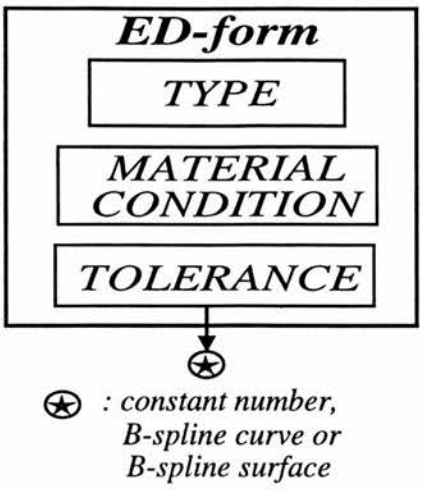


Figure 3.7: The ED-form structure.

- MATERIAL CONDITION: it defines the material condition associated with the tolerance that can be: NONE, M (maximum material condition), L (least material condition) and S (regardless of material condition).
- TOLERANCE: data structure that defines the tolerance zone (see below).
- TYPE: varieties of tolerance (see the possible different varieties in Tables 3.1 and 3.2)

The TOLERANCE data structure defines the size of the tolerance zone that can be CONSTANT (when the tolerance zone presents constant size) or VARIABLE (when the size of the tolerance zone may vary). Variable tolerance zones are defined using B-spline curves or B-spline surfaces depending whether the toleranced feature is a curve or a surface. The variable tolerance zone is defined as the region between the curves or surfaces defining the variable tolerance zone. For instance, in the case of a sculptured surface the variable tolerance zone corresponds to the region between the two B-spline surfaces that define the variable tolerance zone.

The B-splines are used to define a different size of tolerance zone associated to each point in the curve or in the surface being inspected. The purpose of defining variable tolerance zones is to allow the definitions of different sizes of tolerance zones in different regions of a given curve or surface. For instance, variable tolerance zones can be used to define different sizes of tolerance zone to different regions of a sculpture surface

TYPE	DESCRIPTION
NULL	no tolerance defined
STANDARD	interpreted as in the standards
REQUICHA	interpreted according to Requicha's ideas

Table 3.1: Possible varieties of tolerance: ED-position and ED-size.

TYPE	DESCRIPTION
NULL	no tolerance defined
REQUICHA	Rechicha's offset zone
PARALLEL	as in the ANSI standards
ANGULARITY	as in the ANSI standards
PERPENDICULARITY	as in the ANSI standards
STRAIGHTNESS	as in the ANSI standards
FLATNESS	as in the ANSI standards
PROFILE	as in the ANSI standards

Table 3.2: Possible varieties of tolerance: ED-form.

according to the functional requirements associated with each of the different regions of the sculptured surfaces.

The data structures of the EDT models were implemented in C++. The implementation includes some special facilities for creating and manipulating all the data structures described, as well as saving and recovering the structures from files.

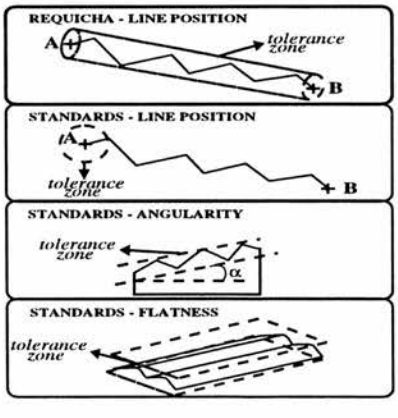


Figure 3.8: Tolerance semantics.

A practical example of the definition of DT nodes can be seen in Appendix D. This appendix contains a list of DT nodes associated with the B-rep model of Appendix C (that corresponds to the B-rep model of the widget in Figure 4.44).

It is important to say that in this section we have just described the syntax we used for attaching tolerances to B-reps. The semantics associated with our implementation is defined in a general way in the ANSI standards and in Requicha's work.

Figure 3.8 illustrates the general meaning of some of the different varieties of tolerances shown in Tables 3.1 and 3.2. A precise definition of a procedural semantics associated with the tolerances of these two tables is given in Chapter 4.

3.4 Conclusions

In this chapter we discussed the problem of representing objects for inspection purposes. This means discussing not only the representation of the nominal shape of objects, but also the problem of representing the allowable variations around the nominal shape of these objects.

It was proposed a set of desirable properties that a representation scheme must have for inspection purposes. The evaluation of these desirable properties was then used to choose the Boundary Representations (B-reps) as the better representational scheme to represent the nominal shape of objects for inspection purposes.

By taking into account the research done in [Johnson 85], we decided to use EDT models for representing the syntax of the tolerance associated with the objects to be inspected. Furthermore, we also extended the syntax of the EDT models to accommodate Requicha's theory of tolerance.

Following these considerations, we built a particular implementation of EDT models that was able to represent the common semantics of tolerancing described in the standards, as well as the semantics associated with Requicha's theory of tolerancing.

A practical example of the construction of an EDT model according to the ideas discussed in this chapter was presented in the Appendices C and D.

It is important to observe that in this chapter we were just concerned with the clear definition of a syntax for representing tolerances associated with the nominal shape of objects. The formal definition of a semantics associated with this syntax is objective of Chapter 4.

Chapter 4

The inspection of common manufactured objects

This chapter presents our approach towards the model based inspection of common manufactured objects whose surface is composed of simple geometric elements such as segments of lines, planes and cylinders.

Section 4.1 describes our objectives and main assumptions. A general overview of our strategy for inspection is given in Section 4.2. Section 4.3 discusses the extraction of some common features as well as the errors made in the extraction process. Section 4.4 proposes a general model of sensitivity and reliability in inspection procedures. A formal model of inspection procedures for the inspection of shape, size and pose is suggested in Section 4.5. Section 4.6 exemplifies the implementation of some inspection procedures using the ideas of Sections 4.4 and 4.5. Our final comments are presented in Section 4.7.

4.1 Introduction

In this chapter we discuss the model based inspection of common manufactured objects whose surface is composed of pieces of planes and cylinders, *i.e.* we are concerned with the inspection of simple geometric elements such as segments of lines, planes and holes.

Our main objective is to present a general approach which allows not only the verification of the most common tolerances defined in the tolerancing standards, but also the inspection of tolerances zones defined according to Requicha's theory of tolerance

modelling. We are also particularly concerned with the evaluation of the designed inspection procedures in order to determine if a designed procedure is suitable for accomplishing a given task and to compare the performance of two different inspection procedures for doing the same task.

As a direct consequence of these issues, throughout this chapter we will:

- propose a general and formal definition of inspection procedures for the verification of the shape, size and pose of simple geometric elements and
- present a general model for the design and the evaluation of the sensitivity and reliability of such inspection procedures.

In the previous chapter we discussed how the EDT models can be used to represent objects for inspection. Under the inspection point of view, an EDT model can be seen as a list of inspection tasks to be accomplished, *i.e.* each node in the DT of the EDT model corresponds to a specific inspection task. Each DT node corresponds to a set of features whose pose, shape or size we want to verify with relation to a set of datum features (that are also indicated by the DT node). In this chapter, we will assume that the inspection of a given object corresponds to the verification of all the DT nodes of the EDT model of the object.

The tolerance defined by a given DT node can only be verified when all the features we want to inspect and the set of datum features are visible by the rangefinder. This means that each DT node can be associated to a restriction in the pose of the object being inspected, *i.e.* the check of a given DT node can only be carried out if the object being inspected presents a pose (in relation to the rangefinder) that satisfies some visibility constraints.

The inspection of a complete EDT model is, thus, related to the satisfaction of a series of viewpoint constraints that are implicitly defined by the list of DT nodes. Therefore, the inspection of an EDT model can only be carried out with the definition of a suitable number of views of the object being inspected, in such a way that each view corresponds to a set of DT nodes to be inspected. Or, in other words, the inspection must be done using a series of known views of the object being inspected.

The use of a series of known views of the object being inspected has its drawbacks:

- the views of the object to be used during the inspection procedure must be defined before the inspection can take place and
- the object being inspected must be accurately manipulated during the inspection.

However, the use of known views of the object being inspected has also its advantages:

- it allows the simulation off-line of the inspection procedures which make it possible to evaluate the sensitivity and reliability of the inspection procedures developed relative to viewpoint and
- it allows a better control of the inspection of the object.

From all that was discussed so far, an important phase in the inspection design is the determination of a global inspection plan that determines the number of different views of the object that will be used during the inspection and the order in which the different features that we want to verify will be inspected. The objective is to make a plan for accomplishing the inspection with the desired reliability, sensitivity and accuracy using the smallest possible number of views of the object we want to inspect.

The complete design of such a planner is outside the scope of this chapter and also of this thesis. In this chapter we are concerned with the design and evaluation of the sensitivity, reliability and accuracy of procedures for the inspection of a given geometric element under a given set of conditions. Figures 4.1 and 4.2 illustrate these ideas.

As we discussed previously, the two first steps in the inspection of mechanical objects using laser stripers are the acquisition of range images of the object being inspected and the registration between a model of the nominal shape of the object and the range image of the actual manufactured object.

We will assume that the object being inspected presents a sufficient number of planar patches on its surface to allow the registration to be done by the segmentation and matching of the planar patches using algorithms based on Faugera's technique (see

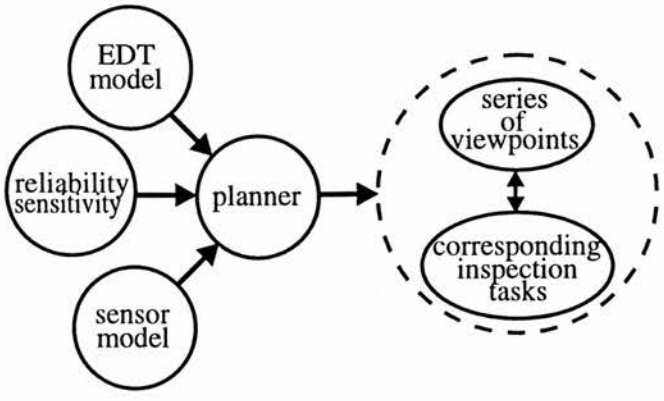


Figure 4.1: Inspection strategy.

Chapter 2). The discussion of the registration process under such conditions is out of the scope of this chapter and an excellent analysis of the accuracy that can be achieved is given in [Marshall 89].

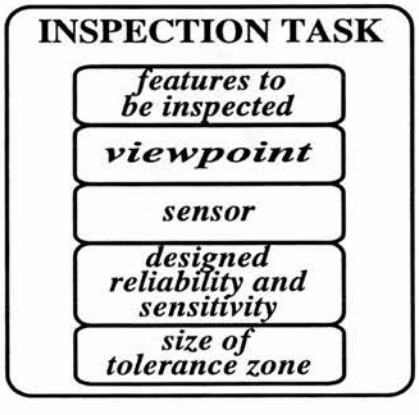


Figure 4.2: The inspection task.

It is important to observe that the accuracy of the registration phase is not critical for the inspection, because the inspection process is only concerned with the verification of the shape, size and relative pose between the geometric elements on the surface of the object being inspected. Therefore, the main objective of the registration process, in this case, is to produce an initial estimate of the position of the geometric elements we want to inspect in the range image.

Having said that, we should say that inspection procedures benefit from increases in the accuracy of the registration process due to better estimates of the initial position of the geometric elements in the range image.

Throughout this chapter we will use the word feature to designate any characteristic of the object that we can inspect. This means that we will use the word feature not only to designate geometric elements in the object's surface such as vertices, edges and holes, but also some relations between these elements such as the angle between two lines or two planes.

In the next section we give a general description of our strategy for the design of inspection procedures.

4.2 General strategy for the definition of inspection procedures

After the data acquisition and registration phases, the inspection of given features requires the comparison of the attributes of these features (shape, size and relative pose to other features) with the allowable variations of these attributes defined in the tolerance specification. In our work we will assume that the allowable variations of the attributes we want to inspect are defined as *tolerance zones* that specify the allowable variations around the nominal value of the attribute. This procedure agrees with the common practices adopted in the standards and Requicha's theory of tolerancing.

As the registration phase involves segmentation and consequently the extraction of features from the range image of the object being inspected, one could think that after registration we could start the inspection immediately. However, this is not true and we must still consider a feature extraction phase due to a few factors:

- the registration does not necessarily involve the extraction of all the features we want to inspect,
- the registration might not extract the features with the desired accuracy and
- in many cases we are interested in extracting features using local properties of the range image which does not happen in many segmentation methods

It follows from our previous discussion that we should consider the extraction of features during inspection procedures. We will, therefore, define an inspection procedure as:

1. Feature extraction

2. Tolerance zones determination (that might involve the determination of datums using some of the features extracted previously)

3. Comparison between the extracted features and their corresponding tolerance zones for determination of the final diagnosis of inspection with chosen values of reliability and sensitivity.

The design of such an inspection procedure will consequently demand the following distinctive phases:

- FEATURE EXTRACTION DESIGN:

  Design/adaptation of feature extraction techniques for the implementation of the inspection procedure. The features we need to extract can be classified as LOCAL features (*e.g.* edges) and GLOBAL features (*e.g.* the angle between two plane normals). The techniques used for the extraction of local features should make the smallest possible number of hypotheses about the object and make use of local operators in order to be able to detect local deformations in the shape (*i.e.* we do not want to extract edge points using the intersection of two planes when we want to inspect the shape of the edge).

  The methods should also be able to improve the original estimate of the feature position (or value) given by the registration and to be robust to the uncertainty in the original estimate of the feature position (or value).

- STATISTICAL ERROR MODELLING:

  To determine the reliability and sensitivity of the inspection procedure of a given feature we need to build a statistical model of the errors in the feature extraction phase of the inspection procedure.

  Ideally, we would like to build theoretical models able to estimate the errors in the feature extraction in all possible circumstances. However, the construction of such models is very complicated. First, the errors in general will depend on a great number of factors such as the sensor errors, the relative position of the

feature in relation to the sensor, the particular characteristics of the feature (*e.g.* in the case of a roof edge the angle between the two planes that make the roof). Besides that, the extraction techniques might involve non-linear operations in which it is very difficult to model the error propagation (*e.g.* thresholding).

Taking into account all these facts we propose to use Monte Carlo simulations for the construction of statistical models. The simulations can be used to build approximate or accurate models. In the first case we use the simulations to obtain rules of thumb for determining the limits in the accuracy of the extraction process. In the second case we use the simulation to build an accurate statistical model under a particular set of conditions.

The use of accurate statistical models is only possible when the whole inspection procedure is planned beforehand. In this case, the inspection should be planned in order to minimise the number of views necessary to carry out all the necessary inspection procedures with the required accuracy. Of course, we will try to give general error models as often as it is possible.

- SENSITIVITY AND RELIABILITY MODELLING:

  Using the statistical model of the errors of the feature extraction algorithms, we can associate with each measured feature f_i^m an uncertainty region R_i such that we have the probability P_i of the actual feature being outside this region. The inspection procedure consists basically in verifying if all the regions R_i corresponding to the measured features being inspected are inside the corresponding tolerance zone. In the case of a global feature f_g, we have just to compare its respective uncertainty region R_g to the allowable variation of the global feature.

  It is possible to consider variations in which we consider a set of measured features out of tolerance only when we have k consecutive regions R_i out of the corresponding tolerance zone. This case corresponds to the case in which we are looking for defects that present at least a given minimum size.

  After we define the uncertainty regions, the determination of the reliability and sensitivity of the inspection procedure is straightforward (see Section 4.4 for a general model of sensitivity and reliability based on tolerance zones). The determination of the uncertainty regions also makes possible the easy verification

of the tolerancing schemes described in the standards and in Requicha's theory of tolerancing.

- TOLERANCE ZONE DETERMINATION:

  After the feature extraction phase, for given values of reliability and sensitivity, all the uncertainty regions R_i corresponding to the set of features we are inspecting are defined. The inspection diagnosis then consists in the verification of the relative position of the uncertainty regions with relation to the corresponding tolerance zone associated with these features.

  Therefore, an important step in the inspection procedure is the determination of the tolerance zone corresponding to the geometric element we want to inspect. This means that we have to define clearly how the shape, scale and relative pose of tolerance zones are determined according to the particular attribute of the geometric element we want to inspect (shape, size or relative pose to other features).

In the next sections we will discuss the details of our ideas concerning the main steps outlined previously, as well as we will show the implementation of some inspection procedures using these ideas.

Section 4.3 discuss the extraction of some common features and the errors made in the extraction of these features. Section 4.4 shows a general model for the sensitivity and reliability of the inspection procedures based in the uncertainty zones. Strategies for determining the shape, scale and pose of tolerance zones are discussed in Section 4.5 that presents a formal definition of inspection procedure diagnoses for the inspection of shape, size and relative pose of geometric elements. Section 4.6 shows some example implementations of inspection procedures for the verification of some of the most common tolerances defined in the tolerancing standards.

4.3 Feature extraction design

As we said before, the initial step in a procedure for inspecting a given geometric element is the extraction of a set of measured features F_m corresponding to this geometric

element. After that, the set of measured features F_m is used by the inspection procedure to determine if the set of actual features produced in the manufacturing process (F_a) presents the desired attributes of size, shape and relative pose to other features.

The errors of the extraction algorithm used (*i.e.* the difference between the set of measured features F_m and the set of actual features F_a) determine the reliability and sensitivity of the inspection procedure. The smaller the errors made by the extraction algorithm, the bigger are the reliability and sensitivity that can be achieved by the inspection procedure.

Ideally, we would like the set of measured features F_m and the set of actual features F_a to be equal, however this is not possible because of the errors already present in the original range image, approximations in the feature extraction algorithm and the numerical errors in the implementation of the extraction algorithm. Although we cannot eliminate the errors in the feature extraction completely, there are a few useful properties that we would like our extraction algorithms to have:

- to have subpixel accuracy,
- to be unbiased, *i.e.* the expected value of the errors of the feature extraction algorithm should be zero,
- to have low probability of producing big errors, *i.e.* the probability of producing a given error of amplitude A should approach zero as the amplitude A increases (the faster the convergence to zero the better),
- to degrade gracefully, *i.e.* the algorithm should be robust as errors in the range image increase,
- to produce bounded errors, *i.e.* even in the presence of big errors in the range image or extremely bad viewing conditions the extraction algorithms should not produce infinite errors,
- to be able to improve the original estimate of the feature position produced by the registration process and

- to be robust to errors in the original estimate of the feature position given by the registration process.

Besides all the above characteristics, as our objective is to use our algorithms to detect local deformations in shape, we also want them:

- to use the smallest possible number of assumptions about the global shape of the object being inspected (*e.g.* we do not want to determine the edge points of a hole using estimates of the hole's radius and center),
- to be as local as possible, *i.e.* the measured features should be extracted using data from the smallest possible neighbourhood around actual feature.

Clearly, it is not possible to satisfy all the properties cited simultaneously. For instance, to increase the robustness to errors in the range image and the accuracy, in general we will have to make the feature extraction less local. Therefore, the design of the feature extraction algorithms consists basically in solving the trade-off created between conflicting objectives in an attempt to accomplish a desired inspection task.

In this section we discuss a few techniques for the extraction of some particular features in range images of common manufactured objects. Our main objective is not only the designing (adaptation) of algorithms but also modelling of the errors made by these algorithms.

As we use Monte Carlos simulations to build statistical descriptions of the errors in the feature extraction, a very important issue is the validity of these statistical descriptions.

The statistical descriptions of the errors are built by applying the feature extraction operators many times to synthetic range images. This means that the correctness of our previsions with respect to the reliability and sensitivity of the inspection procedures that we will design in this chapter depends basically upon two factors:

1. the fidelity with which our synthetic images represent the real range images and
2. the effect of the differences between the synthetic images we generate and the real range images upon the accuracy of our statistical predictions.

Both of these aspects are discussed in Appendix B.

In our simulations the generation of the synthetic images was done by:

1. **Generation of ideal range measurement**: each pixel in the range image is associated with an ideal value of range measurement that is equal to the average height of the surface over the pixel area.

2. **Corruption of ideal measurement with noise**: a normally distributed random process with mean equal to zero and standard deviation equal to σ_S is added to the ideal range measurement.

As we discussed in Appendix B, the use of a normally distributed random process in stage *2* of the generation of the range image is not exactly correct as the actual distribution of the errors in the range measurement is not exactly normally distributed. The effect of this difference in the result of the Monte Carlo simulations is discussed in Appendix B from which we can conclude that the use of the normal distribution in the simulations make it impossible to obtain an accurate estimate of the performance of the extraction algorithms when they are applied to real range data. This means that the use of synthetic images in the simulation allows only the verification of the qualitative behaviour of the algorithms when applied to real range data.

Despite this fact, we used the synthetic images in all the Monte Carlo simulations of Section 4.3. We did so because in Section 4.3 we are not interested in the exact determination of the uncertainty regions associated to the feature extraction algorithm. Our main objective is to illustrate how the Monte Carlo simulations can be used to determining the uncertainty regions associated with the feature extraction operators (more accurate estimates can be obtained by doing a more careful generation of the synthetic images if necessary).

It is important to say that this section does not aim to develop the very best possible algorithm for extracting a given feature, but to develop a set of good algorithms that can be used to demonstrate our ideas towards the inspection of simple geometric elements. The basic structure that we will use for describing the algorithms in the following sections will be:

1. **Objectives and motivation**

   Describes the reasons why the algorithm was implemented.

2. **Algorithm description**

   Describes the algorithm.

3. **Error analysis**

   Analysis of the errors made by the algorithm in the feature extraction. This section aims to illustrate how the uncertainty zones associated with the algorithm can be estimated. In this section the performance of the algorithm is analysed using Monte Carlo simulations in which the algorithm is applied to synthetic images. The number of realisations and the parameters used in the simulations were chosen in an *ad hoc* manner:

   (a) the number of realisations was chosen as big as possible (at least *100*) in such a way that the execution time of the Monte Carlo simulations was not excessively big.

   (b) the parameters used in the simulation were chosen in such a way that the simulation corresponded to the most typical behaviour of algorithm being analysed.

4. **General properties of the algorithm**

   As often as it is possible, we will finish the description with this section in which we describe the main properties that we observed during the Monte Carlo simulations. We will not give any formal proof of these properties as they correspond only to a summary of the results we observed in our simulations.

The algorithms developed in Section 4.3.1 to Section 4.3.3 (extraction of roof edges and step edges) make the assumption that the range image corresponds to a map that associates points in a regular grid in the x and y directions to depth values in the z direction. Furthermore, we also assume that the range image does not present outliers in Section 4.3.1 to Section 4.3.7. These assumptions correspond to the actual behaviour of many rangefinders as, for instance, the Edinburgh's laser striper [Naidu *et al.* 90], where most outliers are removed by their two-camera consistency checking algorithm.

It is important to observe that although the extraction algorithms developed in Section 4.3 will not work when the rangefinder does not satisfy the assumptions made in the previous paragraph, the general strategy developed for modelling the uncertainty regions of the extraction algorithms using Monte Carlo simulations will still be valid. This means that the use of rangefinders that do not satisfy the assumptions made in the previous paragraph will imply only in the development of new feature extraction algorithms that take into account the kind of rangefinder used.

The adaptation of the algorithms developed in Section 4.3 to make them able to cope with outliers in the range image could be done by adding a new processing stage before the actual feature extraction takes place. The objective of the new process stage would be the elimination of the outliers of the range image. An example of the implementation of such algorithm can be seen in [Marshall 89].

The adaptation of the algorithms developed in Section 4.3.1 to Section 4.3.3 (extraction of roof edges and step edges) in order to make them able to cope with non-regular sampling is not possible. This happens because these algorithms were designed to take advantage of the regular sampling in the range image.

The application of all the algorithms described in the next sections for the inspection of objects using real and synthetic images can be seen in Section 4.6.

4.3.1 Extraction of roof edges corresponding to the intersection of two planes

4.3.1.1 Objectives and motivation

In this section our objective is to develop an algorithm for determining the position of the 3D points corresponding to the C_1 discontinuity created by the intersection of two planes when these two planes are visible in the range image (*i.e.* the angle between the plane normals and the viewing direction is smaller than 90°).

Our final intention is to use the extracted 3D edge points for the inspection of segments of line (shape, size and pose) as we will illustrate later in Section 4.6.

4.3.1.2 Algorithm description

We shall call *3D edge line* the 3D line corresponding to the intersection of the two planes and *2D edge line* the projection of this line onto the image plane.

To develop the extraction algorithm we will assume that we have an initial estimate of the 2D edge line (calculated during the registration process) as illustrated in Figure 4.3, and we will also assume that the initial estimate of the 2D edge line in the image plane will correspond to pixels with x coordinate between x_{min} and x_{max}, see Figure 4.3.

Let us now consider a set of N 3D points $P_{edge} = \{p_1, p_2, ..., p_N\}$ belonging to the same column in the range image such that: the x coordinate (x_o) corresponding to this column is bigger than x_{min} and smaller than x_{max} and the original estimate of the 2D edge line passes through the middle of the projection of the set P_{edge} on the image plane (see Figure 4.3).

In this situation we are sure that there is a 3D point with coordinates (x_o, y_o, z_o) belonging to the 3D edge line such that the coordinate y_o is between the coordinate y_1 of p_1 and the coordinate y_N of p_N. Our objective now is to use the set P_{edge} to determine the values of y_o and z_o as accurately as possible.

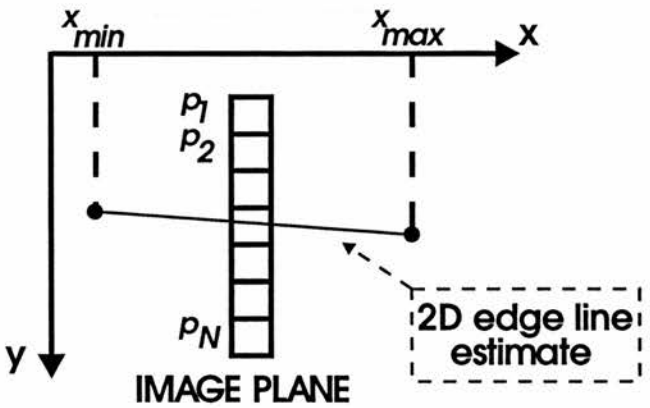


Figure 4.3: Projection of the initial estimate of the 2D edge line on image plane.

In order to achieve our objective, we will assume a parametric model relating the y and z coordinates associated with the pixels in P_{edge}. If we do not consider the sampling errors and the range measurement errors, the y coordinates of the points in P_{edge} and

the range measurements z associated with these pixels can be related by:

$$z = \begin{cases} k_1(y - y_o) + z_o & y \geq y_o \\ k_2(y - y_o) + z_o & y < y_o \end{cases} \tag{4.3.1}$$

where: k_1 and k_2 are constants that depend on the relative position between the intersecting planes and the range sensor.

For a given value of y_o we can determine optimal values of k_1^*, k_2^* and z_o^* such that the average distance d_{AV} (see Equation (4.3.2)) between P_{edge} and the parametric model given by Equation (4.3.1) is minimum.

$$d_{AV} = \frac{\sum_{P_{small}} (z_i - k_2(y_i - y_o) - z_o)^2 + \sum_{P_{big}} (z_i - k_1(y_i - y_o) - z_o)^2}{N_s + N_b} \tag{4.3.2}$$

where:

- z_i is the range measurement (z coordinate) of the pixel p_i in P_{edge}
- y_i is the y coordinate of the pixel p_i in P_{edge}
- P_{small} is a subset of P_{edge} with N_s 3D points such that all the 3D points in P_{small} have y coordinates smaller than y_o
- P_{big} is a subset of P_{edge} with N_b 3D points such that all the 3D points in P_{big} have y coordinates bigger than y_o

The values of k_1^*, k_2^* and z_o^* that minimise d_{AV} in Equation (4.3.2) can be calculated by solving the linear system of equations given by Equations (4.3.3) to (4.3.5):

$$\sum_{P_{small}} z_i(y_i - y_o) = k_1^* \sum_{P_{small}} (y_i - y_o)^2 + z_o^* \sum_{P_{small}} (y_i - y_o) \tag{4.3.3}$$

$$\sum_{P_{big}} z_i(y_i - y_o) = k_2^* \sum_{P_{big}} (y_i - y_o)^2 + z_o^* \sum_{P_{big}} (y_i - y_o) \tag{4.3.4}$$

$$\sum_{P_{small}+P_{big}} z_i = k_1^* \sum_{P_{small}} (y_i - y_o) + k_2^* \sum_{P_{big}} (y_i - y_o) + z_o^*(N_s + N_b) \tag{4.3.5}$$

Using Equations (4.3.3) to (4.3.5) and (4.3.2) we can define the function $f_{distance}(y)$ that associates with each value y the minimum average distance, defined by Equation (4.3.2), between the set of pixels P_{edge} and the parametric model of a roof edge (defined by Equation (4.3.1)) with a vertex at x_0 and y.

Therefore, the determination of the 3D edge point corresponding to the set of pixels P_{edge} can be defined as finding the value y_o^* of y that minimises the function $f_{distance}$. The value of z_o^* found during the evaluation of $f_{distance}(y_o^*)$ is the z coordinate of the 3D edge point.

The minimisation of $f_{distance}$ is a non-linear process that we solved by adopting a multiscale search in which we start the search in a neighbourhood of our original estimate of y_o and reduce the size of the neighbourhood in which we look for the minimum of $f_{distance}$ at each stage. The search is stopped when the variation in the estimate of y_o is smaller than a chosen constant ϵ_{conv}. This whole process can be described as:

1. start with an original estimate of the y coordinate of the edge point ($\hat{y_o}$), a value of increment (δy) and an integer number M
2. built the set $Y = \{y^{-M}, ..., y^0, ..., y^M\}$ such that $y^0 = \hat{y_o}$ and $y^i - y^{i-1} = \delta y$
3. find the value of y^k in Y corresponding to the minimum value of $f_{distance}$
4. If $\delta y \leq \epsilon_{conv}$ make $\hat{y_0} = y^k$ and stop the process. Otherwise make $\hat{y_0} = y^k$, $\delta y = \frac{\delta y}{M}$ and go to step **2**.

In the minimisation process described above the original estimate of y_o and the initial value of δy are chosen by considering the uncertainty in the initial edge estimate. Typically, we start the minimisation with $\hat{y_o}$ in the middle of P_{edge}, δy equal to the y size of the pixel in the range image and M equal to 10.

The whole process described so far, will not work in the cases in which the 2D edge line is parallel (or almost parallel) with the y axis of the laser striper reference frame, because in this case the whole set P_{edge} could be made of points belonging to the 3D edge line.

This problem can be easily solved if we consider the angle θ between the estimate of the 2D edge line given by the registration and the line corresponding to the y axis. For values of θ bigger than 45° we will adopt exactly the same procedure we have described. However, when θ is smaller than 45° we will adopt a similar procedure in

which P_{edge} is a set of pixels belonging to the same row in the range image. In this case, we will minimise the function $f_{distance}$ to find an estimate of the x coordinate of a 3D point belonging to the 3D edge line.

The value of the range measurement associated with a pixel when the 2D edge line falls in the middle of this pixel might not follow the parametric model described by Equation (4.3.1). Because of this fact, we define the sets P_{small} and P_{big} by eliminating the pixel corresponding to our current estimate of y_o (x_o if θ is smaller than 45°) and its two surrounding pixels.

4.3.1.3 Error analysis

The final error in the estimates of the 3D edge points can vary a great deal according to a considerable number of factors:

1. the relative position between the 3D edge line and the image plane,
2. the angles between the two intersecting planes and the viewing direction,
3. the range measurement errors of the laser striper used,
4. the size of the pixels in the range image and
5. the number of points in P_{small} (N_s) and P_{big} (N_b)

The two first items in the list above are related to the shape of the roof edge and to its position relative to the sensor. The smaller errors correspond to cases in which the absolute value of $k_1 - k_2$ is big (*i.e.* we have a very distinctive roof edge that cannot be confused with a line).

Also, smaller errors in the estimates of z_o occur with planes whose normals make angles near 0° with the viewing direction (for angles near 90° we have big variations in z_o for small variations in the estimate of the coordinates x and y of the 3D edge point).

The third and fourth items in the list are related to the accuracy and precision of the laser striper sensor. In our simulation analysis we will assume that the errors in the range measurement of the laser striper are normally distributed, with zero mean and

standard deviation σ_S. The errors in the 3D edge point estimates decrease when the size of the pixel in the range image and the value of σ_S decrease.

Finally, the fifth item is related to the particular characteristics of the extraction algorithm. As the algorithm is based on the use of the average distance d_{AV} between a parametric model and the set P_{edge}, the errors will decrease when N_s and N_b increase.

Finding an accurate theoretical statistical model of the uncertainty in the determination of the 3D edge points is extremely complicated: the errors depend upon a considerable number of parameters and the extraction involves the non-linear minimisation of $f_{distance}$. Therefore, we proceeded with Monte Carlo simulations in which we generated synthetic range images for different angles between the intersecting planes and viewing conditions, and observed the errors made by the extraction algorithm.

The synthetic images were created by calculating the ideal range measurement associated with each pixel in the range image and adding to this value a normally distributed random variable of zero mean and standard deviation σ_S simulating the laser striper errors. The ideal range measurement associated with each pixel p in the range image was calculated by averaging the exact range over the portion of the scene projected to pixel p.

As the algorithm finds the y (or x) and z coordinates of an edge point with a given x (or y) coordinate, we will have errors in just two of the three coordinates of the 3D edge point and the uncertainty region associated with each edge point is a two dimensional figure, *i.e.* the uncertainty regions are subsets of the plane yz (or xz) of the sensor reference frame.

Tables 4.1, 4.2, 4.3 and 4.4 show the maximum absolute errors and the standard deviation of the errors in the estimate of y_o and z_o for different angles ϕ_1 and ϕ_2 between the normals of the intersecting planes and the viewing direction.

Each element in the tables was obtained by considering 100 different realisations of the synthetically created images. As for each pair of angles ϕ_1 and ϕ_2 there are an infinite number of different 3D edge lines (corresponding to the different poses of the 3D edge line in the sensor reference frame), in each realisation the particular 3D edge line considered was generated randomly. In all the realisations we used $N_s = N_b = 18$

		$\phi_2 = 0.16$ rad	$\phi_2 = 0.47$ rad	$\phi_2 = 0.79$ rad	$\phi_2 = 1.1$ rad
$\phi_1 = 0.0$ rad		1.994	0.765	0.252	0.174
$\phi_1 = 0.31$ rad		1.930	2.161	0.325	0.201
$\phi_1 = 0.63$ rad		0.530	1.291	0.940	0.239
$\phi_1 = 0.94$ rad		0.199	0.298	0.666	0.431

Table 4.1: Maximum absolute error in the estimate of y_0 (in mm) for different angles ϕ_1 and ϕ_2 between the intersecting planes and the viewing direction. Result of 1000 realisations for each pair of values of ϕ_1 and ϕ_2. Relative pose of 3D edge lines with relation to sensor created randomly. Synthetic images created considering pixels of $0.5\ mm \times 0.5\ mm$ and standard deviation of range measurement in laser striper equal to $0.1mm$. Edge extraction with $N_s = N_b = 18$.

which corresponds to a length of $9\ mm$ if we consider pixels of $0.5\ mm \times 0.5\ mm$.

As a first comment on the results of the simulation, in most of the cases considered the average error in the estimate of y_o and z_o was approximately equal to zero (at least one order of magnitude smaller than the standard deviation of the error in most of the cases) which means that the algorithm was unbiased during almost all the simulations.

As we can see from the tables, the performance of the algorithm can vary a great deal according to the particular values of the angles ϕ_1 and ϕ_2 and the viewing conditions. Although the table shows that the algorithm has a good performance for a considerable number of situations, it is important to observe that the final performance of the algorithm can degrade a great deal. It is also important to understand that Tables 4.1 and 4.2 show the very worst performance of the algorithm during the simulations, *i.e.* these tables show the errors of the algorithm under the worst viewing conditions found during the simulation(pose of the 3D edge line).

For instance, the extraction can be very inaccurate when the slope of the intersecting planes and the viewing conditions are such that the original scene roof edge projects onto the range image as a curve very close to an actual straight line. In this case, we can only obtain reasonable accuracy by increasing the number of pixels in P_{edge}, and the algorithm might even have an average error different than zero (with the same order of magnitude as the standard deviation of the error).

It is important to observe that the simulations show the performance of the algorithm in a series of very tough tests. This happens because the maximum errors occur in

	$\phi_2 = 0.16$ rad	$\phi_2 = 0.47$ rad	$\phi_2 = 0.79$ rad	$\phi_2 = 1.1$ rad
$\phi_1 = 0.0$ rad	0.083	0.152	0.121	0.185
$\phi_1 = 0.31$ rad	0.430	0.532	0.241	0.310
$\phi_1 = 0.63$ rad	0.254	0.415	0.409	0.281
$\phi_1 = 0.94$ rad	0.200	0.423	0.627	0.384

Table 4.2: Maximum absolute error in the estimate of z_0 (in *mm*) for different angles ϕ_1 and ϕ_2 between the intersecting planes and the viewing direction. Result of 1000 realisations for each pair of values of ϕ_1 and ϕ_2 under the same conditions of Table 4.1.

	$\phi_2 = 0.16$ rad	$\phi_2 = 0.47$ rad	$\phi_2 = 0.79$ rad	$\phi_2 = 1.1$ rad
$\phi_1 = 0.0$ rad	0.508	0.153	0.078	0.043
$\phi_1 = 0.31$ rad	0.333	0.293	0.086	0.043
$\phi_1 = 0.63$ rad	0.110	0.245	0.176	0.052
$\phi_1 = 0.94$ rad	0.059	0.070	0.143	0.088

Table 4.3: Standard deviation of error in the estimate of y_0 (in *mm*) for different angles ϕ_1 and ϕ_2 between the intersecting planes and the viewing direction. Result of 1000 realisations for each pair of values of ϕ_1 and ϕ_2 under the same conditions of Table 4.1.

	$\phi_2 = 0.16$ rad	$\phi_2 = 0.47$ rad	$\phi_2 = 0.79$ rad	$\phi_2 = 1.1$ rad
$\phi_1 = 0.0$ rad	0.042	0.071	0.052	0.080
$\phi_1 = 0.31$ rad	0.123	0.110	0.076	0.053
$\phi_1 = 0.63$ rad	0.058	0.137	0.132	0.189
$\phi_1 = 0.94$ rad	0.073	0.124	0.198	0.130

Table 4.4: Standard deviation of error in the estimate of z_0 (in *mm*) for different angles ϕ_1 and ϕ_2 between the intersecting planes and the viewing direction. Result of 1000 realisations for each pair of values ϕ_1 and ϕ_2 under the same conditions of Table 4.1.

situations in which the original roof edges project on the range image as curves that are practically a straight line. The edges corresponding to the maximum absolute errors in Tables 4.1 and 4.2 are from data sets that can be easily taken by actual straight lines. For instance, the least square approximation of these curves as straight lines have a correlation coefficient that is always bigger than 0.87. Figure 4.4 illustrates this fact.

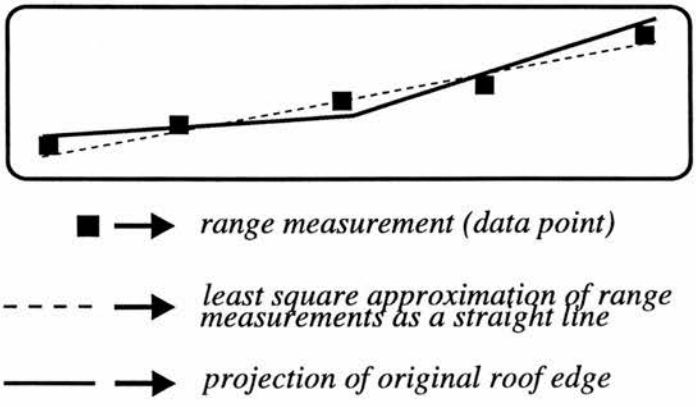


Figure 4.4: Range measurements corresponding to roof edge, projection of original roof edge and least square approximation of range measurements as a straight line.

In order to understand the variation of the errors with the number of pixels used in the extraction procedure ($N_{pixel} = N_s = N_b$) let us consider the variation of the errors with N_{pixel} in a particular situation.

Figures 4.5 and 4.6 show the maximum error, the standard deviation, the average error, the standard deviation multiplied by minus one and the minimum errors made in the estimate of the y_o and z_o coordinates of the edge position for different values of N_{pixel}.

The results were obtained in *300* simulations of the extraction of the roof edges for each value of N_{pixel} for two intersecting planes whose normals make angles $\phi_1 = 0.8$ *rad* and $\phi_2 = 1.0$ *rad* with the viewing direction in such a way that the 2D edge line makes an angle of 45° with the x direction and the 3D edge line makes an angle of *0.8 rad* with the image plane.

The results illustrate how the errors decrease when we increase the number of pixel points N_{pixel} and the detection algorithm becomes less and less local. The graphs in Figures 4.5 and 4.6 can be used to determine an upper limit on the size of the uncertainty regions associated with the extraction algorithm under a given set of conditions.

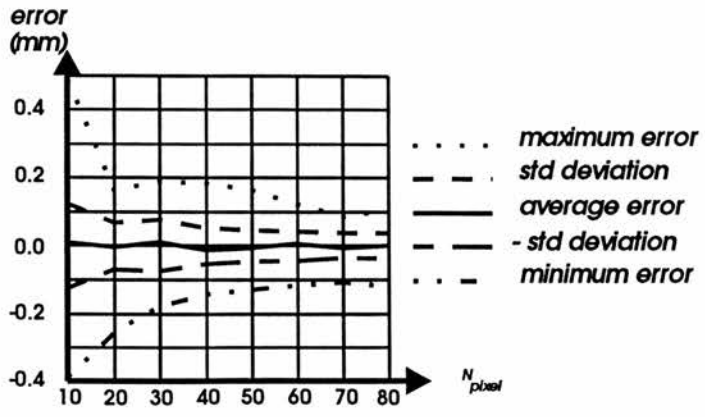


Figure 4.5: Maximum error, standard deviation, average error, minus standard deviation and minimum error of estimate of y_o. Result of 300 simulations to each value of N_{pixel} for $\sigma_S = 0.1$ *mm*, $\phi_1 = 0.8$ *rad*, $\phi_2 = 1.0$ *rad*, 2D edge line making angle of 45° with x direction, 3D edge line making angle of *0.8 rad* with the image plane and N_{pixel} varying from 10 to 80 pixels.

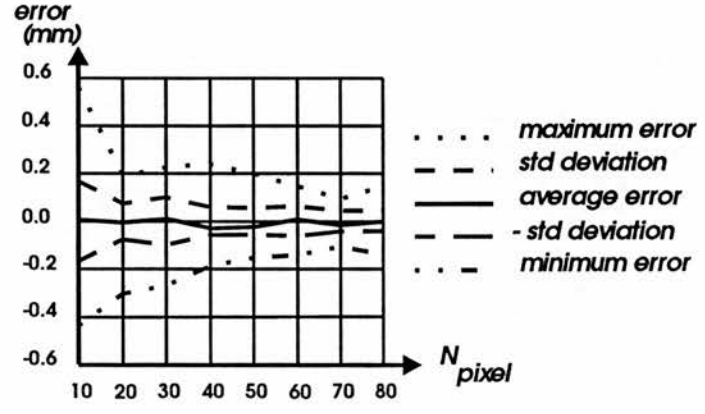


Figure 4.6: Maximum error, standard deviation, average error, minus standard deviation and minimum error of estimate of z_o. Result of 300 simulations under the same conditions of Figure 4.5.

For instance, for $N_{pixel} = 20$ the probability of the absolute difference between the actual y_o coordinate and its measured value being bigger than *0.25 mm* is very low. As will be discussed in Section 4.4, the size of the uncertainty regions is associated with the size of the tolerance zones we can inspect. This means that these graphs allow the determination of the number of pixels N_{pixel} necessary to inspect a given size of tolerance zone with a high value of sensitivity and reliability (under a given set of circumstances).

In order to obtain a more precise estimate of the uncertainty regions associated with the 3D edge points extracted by the algorithm, we will approximate the uncertainty regions by 2D regions limited by ellipses. This procedure corresponds to the approximation

of the error in the edge extraction by two normally distributed random variables with mean equal to zero.

Figure 4.7 shows the variances of the error in the estimate y_o and z_o as well as the value of the covariance between the errors in these two coordinates obtained in the same simulation conditions of Figures 4.5 and 4.6. These graphs were obtained by applying Equations (4.3.6) to (4.3.8) to the result of the Monte Carlo simulation.

If we apply Equations (4.3.9) to (4.3.11) to the results shown in Figure 4.7 we can calculate the main axis a_z and b_y of the ellipse that delimits the uncertainty region when we approximate the errors by normally distributed random variables. These values of a_z and b_y correspond to a probability approximately equal to 0.61 of the actual values of y_o and z_o being outside the uncertainty region (according to the normal approximation). Figure 4.8 shows the values of a_z and b_y corresponding to the data in Figure 4.7.

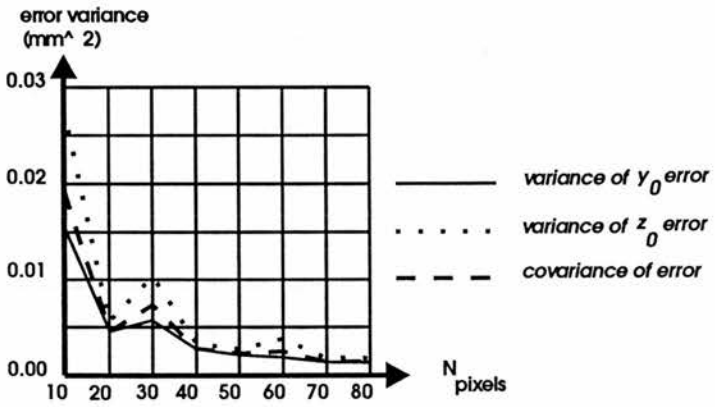


Figure 4.7: Variance of the error in the estimate of y_o, variance of the error in the estimate of z_o and covariance between the errors in the estimates of y_o and z_o. Result of 300 simulations under the same conditions of Figure 4.5.

$$\hat{var}_{yy} = \frac{1}{N_T - 1} \sum_{i=1}^{N_T} (\hat{y_o}^i - y_o)^2 \tag{4.3.6}$$

$$\hat{var}_{zz} = \frac{1}{N_T - 1} \sum_{i=1}^{N_T} (\hat{z_o}^i - z_o)^2 \tag{4.3.7}$$

$$\hat{cor}_{yz} = \frac{1}{N_T - 1} \sum_{i=1}^{N_T} (\hat{y_o}^i - y_o)(\hat{z_o}^i - z_o) \tag{4.3.8}$$

$$\alpha = \frac{1}{2} atan\left(\frac{-2co\hat{r}_{yz}}{va\hat{r}_{yy} - va\hat{r}_{zz}}\right) \tag{4.3.9}$$

$$a_z = \sqrt{\frac{va\hat{r}_{yy} + va\hat{r}_{zz} + \frac{2co\hat{r}_{yz}}{sin(2\hat{\alpha})}}{2}} \tag{4.3.10}$$

$$b_y = \sqrt{\frac{va\hat{r}_{yy} + va\hat{r}_{zz} - \frac{2co\hat{r}_{yz}}{sin(2\hat{\alpha})}}{2}} \tag{4.3.11}$$

where:

- N_T is the total number of simulations,
- $\hat{y_o}^i$ and $\hat{z_o}^i$ are estimates of the actual edge coordinates (y_o and z_o) found by the extraction algorithm in iteration i,
- a_z and b_y are the main axis lengths of the uncertainty ellipse corresponding to the approximation of the errors in the extraction by a normally distributed random vector and
- α is the angle between the estimated uncertainty ellipse and the z axis of the sensor reference frame.

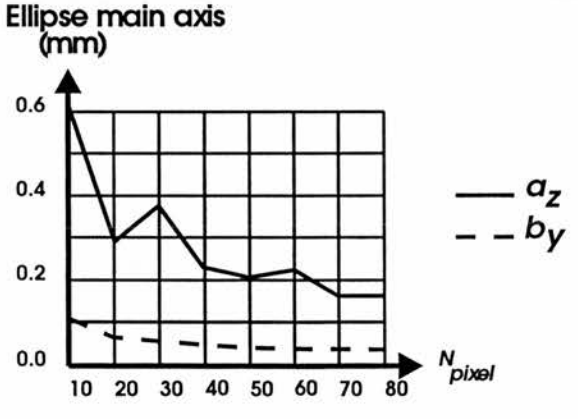


Figure 4.8: Main axis of ellipse limiting uncertainty region of extraction algorithm. Result of 300 simulations under the same conditions of Figure 4.5.

Let us now consider an uncertainty region limited by an ellipse centered on $\hat{y_o}$ and $\hat{z_o}$, making an angle α with the z axis of the sensor, such that its main axes are of size $K_0 a_z$ and $K_0 b_y$ (where K_0 is a real number). See Figure 4.9.

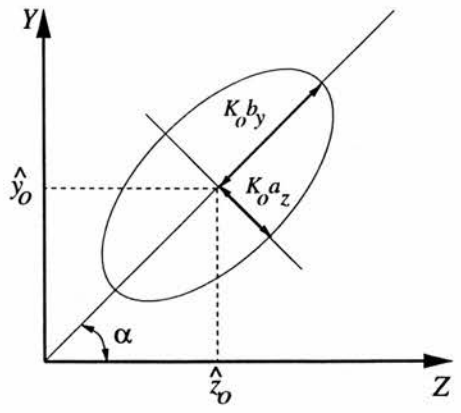


Figure 4.9: Uncertainty region illustration.

Let us also assume that α, a_z and b_y were obtained by applying Equations (4.3.6) to (4.3.11) to the result of Monte Carlo simulations.

In this case, the estimate $\hat{P}_i$ of the probability P_i of the actual values of y_o and z_o being outside of the uncertainty region is given by:

$$\hat{P}_i = e^{\frac{-K_0{}^2}{2}} \tag{4.3.12}$$

The normal estimate $\hat{P}_i$ is not a good estimate of the actual probability P_i, as $\hat{P}_i$ will usually underestimate P_i (see Figure 4.10).

To correct this fact, we will scale the dimensions of the uncertainty ellipse by a constant factor K_{scale} bigger than one. After the scaling, the new normal approximation of the probability P_i is given by:

$$\hat{P}_i = e^{-\frac{1}{2}(\frac{K_0}{K_{scale}})^2} \tag{4.3.13}$$

The constant K_{scale} in Equation (4.3.13) will be determined considering the result of the Monte Carlo simulation in such a way that the value of $\hat{P}_i$, given by Equation (4.3.13), is equal to the value of P_i obtained in the Monte Carlo simulation for a desired value of P_i. Figure 4.11 shows the different values of K_{scale} obtained for different values of P_i in the same simulation conditions as Figure 4.8.

4.3.1.4 General properties of algorithm

In all the discussion done so far, we have been concerned with the determination of the uncertainty regions associated with the extraction algorithm under a given set of

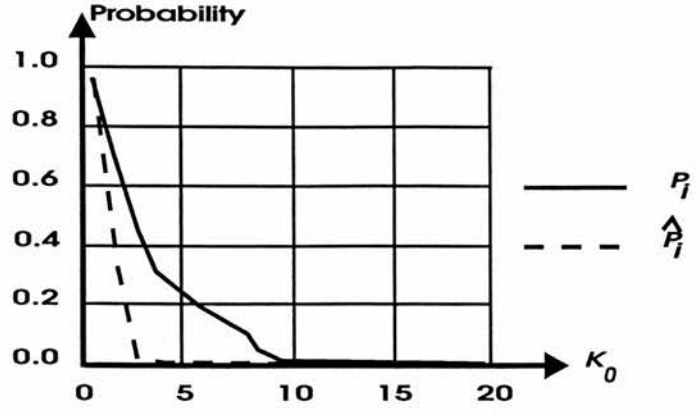


Figure 4.10: Probability P_i and normal estimate $\hat{P}_i$ of probability P_i. Result of 300 simulations under the same conditions as Figure 4.5 when N_{pixel} is equal to 40 pixels.

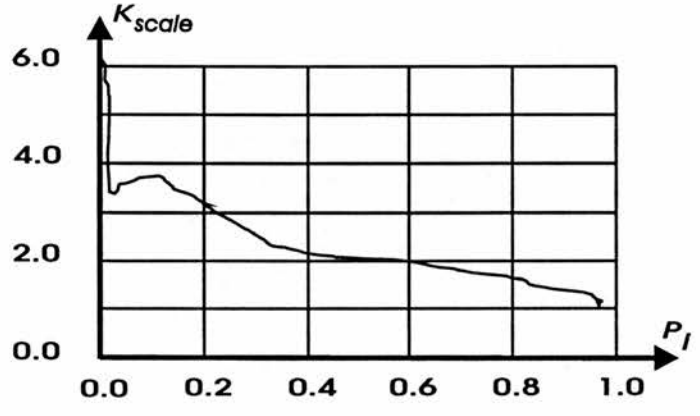


Figure 4.11: Variation of K_{scale} with P_i. Result of 300 simulations under the same conditions of Figure 4.5 when N_{pixel} is equal to 40 pixels.

conditions. As we explained before, the determination of an accurate model of the uncertainty regions valid in all possible circumstances is very complicated. However, the results of our simulations showed some general properties of the uncertainty regions:

1. For a given angle between the intersecting planes and the viewing direction, size of pixel and number of pixels used in the extraction (N_b and N_s), the error in the extraction of the 3D edge points varies approximately linearly with the standard deviation of the error in the range measurement σ_S (for values of σ_S between *0.01 mm* and *1 mm*). This means, for instance, that the standard deviation of the errors obtained with a sensor with σ_S equal to *0.01 mm* will be approximately ten times smaller than the standard deviation of the errors obtained when σ_S is equal to *0.1 mm*. This fact is illustrated by Figure 4.12 that shows the variation

of the standard deviation of the errors in the estimate of y_0 with σ_S. This figure was obtained as a result of 500 simulations for each value of σ_S in the same conditions of Figure 4.5 and with $N_b = N_s = 40$.

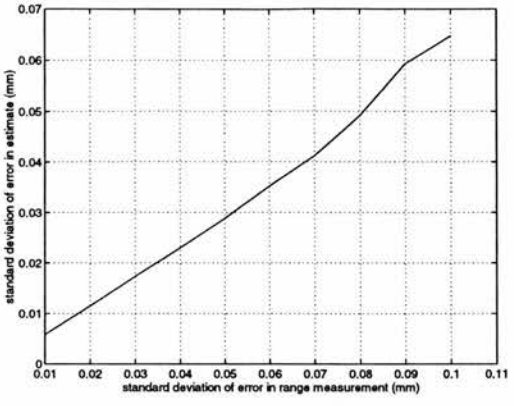


Figure 4.12: Standard deviation of error in the estimate of y_0 versus the standard deviation of error in range measurement σ_S. Result of 500 simulations for each value of σ_S under the same conditions of Figure 4.5 and with $N_b = N_s = 40$.

2. For planes whose normals make an angle smaller than approximately *1.1 rad* with the viewing direction and errors in the estimate of y_0 smaller than *0.3 mm*, the errors in the estimate of z_0 will also be smaller than approximately *0.35 mm*. This fact is illustrated by Tables 4.1 and 4.2 in which we can observe that errors smaller than *0.3 mm* in the estimate of y_0 in Table 4.1 correspond to errors smaller than approximately *0.35 mm* in the estimate of z_0 in Table 4.2.

3. The performance of the algorithm varies enormously with the position of the 3D edge line with respect to the sensor reference frame. The best accuracy is achieved when the 3D edge line is parallel to the image plane. This fact is illustrated by Figure 4.13 that shows the variation of the maximum error in the estimate of y_0 with the angle between the 3D edge line and the image plane. The graph was obtained as a result of simulations under the same conditions of Figure 4.5 but with different values of angles between the 3D edge line and the image plane. To each value of angle 500 simulations were considered.

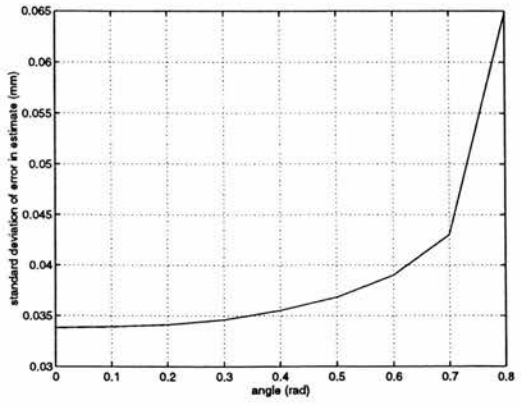


Figure 4.13: Standard deviation of error in the estimate of y_0 versus the angle between 3D edge line and image plane. Result of 500 simulations for each value of angle under the same conditions of Figure 4.5 and with $N_b = N_s = 40$.

4.3.2 Extraction of step edges corresponding to the intersection of two planes

4.3.2.1 Objectives and motivations

Our objective now is the determination of the position of 3D step edge points corresponding to C_0 discontinuities in the range image that appear due to the intersection between two planes when one of the two intersecting planes is not visible in the range image. As before, we will use the names 3D edge line and 2D edge line to designate the two lines corresponding to the C_0 discontinuity and their projection on the image plane.

The extracted 3D step edge points will be used in the inspection of the length of segments, as well as the inspection of holes (shape, size and pose).

4.3.2.2 Algorithm description

To extract the 3D step edge points we will use the same idea we used for the extraction of roof edges in the previous section and we will try to fit a parametric model of the step edge to the range image. We will also only consider the cases in which the surfaces on the two sides of the 2D edge line are planes (*i.e.* there are 3 planes involved: the front surface plane, the invisible back-facing plane and the more distant plane seen on the other side of the edge).

In the same way as we did in the previous section, we will use the original estimate of the 2D edge line given by the registration (see Figure 4.3). As before, we will define a set of points $P_{edge} = \{p_1, p_2, ..., p_N\}$ in which all the points belong to the same column with x coordinate equal to x_o. Furthermore, we will build P_{edge} in such a way that our original estimate of the 2D edge line passes in the middle of P_{edge} (see Figure 4.3).

Under these circumstances, there must be a 3D step edge point at coordinates (x_o, y_o, z_o) such that y_o has a value between the y coordinate of p_1 and the y coordinate of p_N.

We will determine the value of the coordinates y_o and z_o through the minimisation of the distance between the set P_{edge} and a parametric model of the step edge given by Equation (4.3.14).

$$z = \begin{cases} k_1(y - y_o) + z_o + step & y \geq y_o \\ k_2(y - y_o) + z_o & y < y_o \end{cases} \tag{4.3.14}$$

where: k_1, k_2 and $step$ are constants that depend on the relative position between the intersecting planes, the laser striper and the plane whose projection on the image plane is limited by the 2D edge line.

For a given value of y_o we can calculate the optimal values k_1^* of k_1, k_2^* of k_2, z_o^* of z_o and $step^*$ of $step$ that minimise the average distance d_{AV} (see Equation (4.3.15)) between P_{edge} and the parametric model given by Equation (4.3.14). These optimal values correspond to the solution of Equations (4.3.16) to (4.3.19).

$$d_{AV} = \frac{\sum_{P_{small}} (z_i - k_2(y_i - y_o) - z_o)^2 + \sum_{P_{big}} (z_i - k_1(y_i - y_o) - z_o - step)^2}{N_s + N_b} \tag{4.3.15}$$

$$\sum_{P_{small}} z_i = z_o^* N_s + k_2^* \sum_{P_{small}} (y_i - y_o) \tag{4.3.16}$$

$$\sum_{P_{small}} z_i(y_i - y_o) = z_o^* \sum_{P_{small}} (y_i - y_o) + k_2^* \sum_{P_{small}} (y_i - y_o)^2 \tag{4.3.17}$$

$$\sum_{P_{big}} z_i = z_o^* N_b + step^* N_b + k_1^* \sum_{P_{big}} (y_i - y_o) \tag{4.3.18}$$

$$\sum_{P_{big}} z_i(y_i - y_o) = (z_o^* + step^* + k_1^*) \sum_{P_{big}} (y_i - y_o) \tag{4.3.19}$$

where: z_i, y_i, P_{small}, N_s, P_{big} and N_b are the same as in Section 4.3.1.

Using Equations (4.3.15) and (4.3.16) to (4.3.19) we can define a function $f_{distance}(y)$ that associates with each value y the minimum average distance d_{AV} between the set P_{edge} and the step edge defined by Equation (4.3.14) when $y_o = y$. Therefore, as in the previous section, the determination of the 3D step edge point (x_o, y_o, z_o) can be understood as the calculation of the coordinate y^* that minimises $f_{distance}$ (notice that the estimated value of z_o will be equal to the value of z_o^* found during the evaluation of $f_{distance}(y^*)$)

As in Section 4.3.1, the minimisation of $f_{distance}$ is a non-linear process. Furthermore, due to the discrete nature of P_{edge} and the existence of a C_0 discontinuity, there is more than one value of the y coordinate that will minimise $f_{distance}$. This fact is illustrated by Figure 4.14 that shows two different solutions with $y_o = y_1$ and $y_o = y_2$.

The existence of multiple solutions happens because it is possible to vary the y coordinate of the 3D edge point between two pixels p_i and p_j without changing the value of d_{av}, *i.e.* the function $f_{distance}$ is a staircase function with steps of the size of the pixel of the range image in the y direction.

Figure 4.15 shows the functions $f_{distance}$ corresponding to a step edge and a roof edge. The graph corresponding to the roof edge was obtained in one of the simulations used to build Figure 4.5. The graph corresponding to the step edge was obtained in the same simulation conditions as Figure 4.5, the only difference being the existence of a C_0 discontinuity of size *3 mm*.

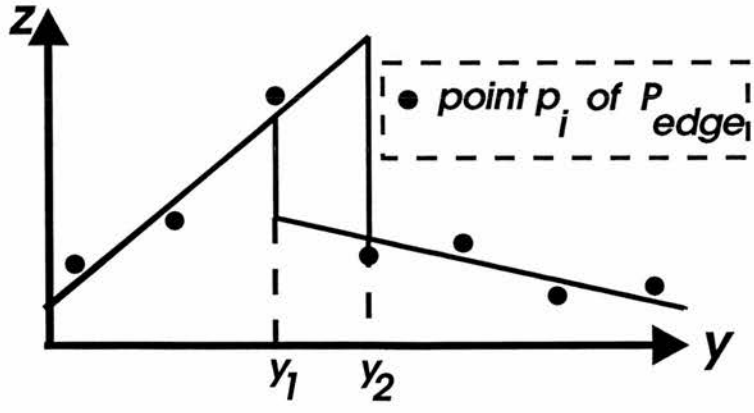


Figure 4.14: Multiple solutions in the minimisation of $f_{distance}$ during the extraction of 3D step edge points.

Taking into account the shape of $f_{distance}$, it does not make sense to try to search for

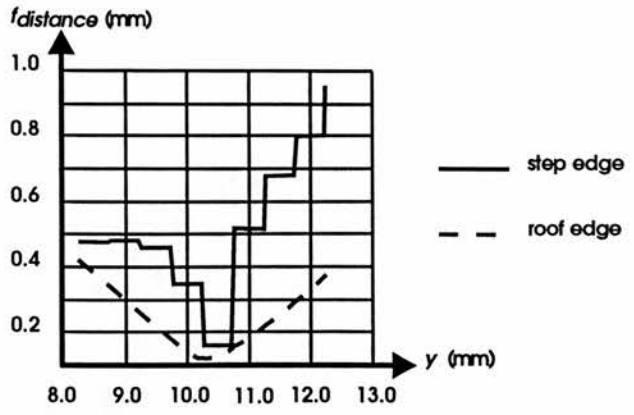


Figure 4.15: Functions $f_{distance}$ obtained in the extraction of a roof edge and a step edge. Result of simulation with under the same conditions of Figure 4.5 with $N_{pixel} = 20$ and C_0 discontinuity of *3 mm* in the step edge.

a minimum of $f_{distance}$ with subpixel accuracy. Therefore, we defined the value of y_o just by evaluating $f_{distance}$ in values of y corresponding to the middle between two neighbour pixels p_i and p_{i+1} in P_{edge}.

As in Section 4.3.1 (and for the same reasons), in the implementation of the extraction algorithm we:

1. considered the angle θ between the 2D edge line estimated by the registration and the line corresponding to the y direction. For values of θ bigger than 45° we proceeded exactly as described so far, and for values of θ smaller than 45° we built the set P_{edge} using points belonging to a same row of the range image. In this case, we minimised $f_{distance}$ to find the x coordinate of the edge pixel.

2. defined the sets P_{big} and P_{small} without considering the pixel corresponding to our current estimate of y_0 (x_0 if θ is smaller than 45°).

4.3.2.3 Error analysis

As before, the exact theoretical modelling of the errors in the extraction of the 3D step edge points is very complex as the errors will depend on a considerable number of parameters related to the shape of the step edge, the accuracy of the sensor and particular characteristics of the algorithm (N_{pixel}).

The case of 3D step edges is further complicated by the fact that the algorithm is biased,

		$\phi_2 = 0.16$ rad	$\phi_2 = 0.47$ rad	$\phi_2 = 0.79$ rad	$\phi_2 = 1.1$ rad
$\phi_1 = 0.0$ rad		0.062	0.087	0.102	0.238
$\phi_1 = 0.31$ rad		0.057	0.058	0.158	0.243
$\phi_1 = 0.63$ rad		0.054	0.126	0.181	0.302
$\phi_1 = 0.94$ rad		0.205	0.178	0.103	0.330

Table 4.5: Standard deviation of error in the estimate of z_0 (in *mm*) for different angles ϕ_1 and ϕ_2 between the visible planes surrounding the step edge and the viewing direction. Result of 100 realisations for each pair of angles ϕ_1 and ϕ_2. Relative pose of 3D edge lines with relation to sensor created randomly. Synthetic images created considering pixels of *0.5 mm* × *0.5 mm* and standard deviation of range measurement in laser striper equal to *0.1 mm*. Edge extraction with $N_s = N_b = 18$. Step of *5 mm* between 3D edge lines at edge point extracted.

		$\phi_2 = 0.16$ rad	$\phi_2 = 0.47$ rad	$\phi_2 = 0.79$ rad	$\phi_2 = 1.1rad$
$\phi_1 = 0.0$ rad		0.164	0.285	0.255	0.939
$\phi_1 = 0.31$ rad		0.112	0.172	0.435	0.717
$\phi_1 = 0.63$ rad		0.172	0.417	0.623	0.585
$\phi_1 = 0.94$ rad		0.671	0.587	0.242	0.961

Table 4.6: Maximum absolute error in the estimate of z_0 (in *mm*) for different angles ϕ_1 and ϕ_2 between visible planes surrounding step edge and viewing direction. Result of 100 realisations for each pairs of angle ϕ_1 and ϕ_2 under the same conditions as Table 4.5.

i.e. for a given set of conditions the expected value of the error in the determination of the 3D edge point is not necessarily equal to zero.

Table 4.6 shows the maximum absolute errors in the estimate of z_0 for different angles ϕ_1 and ϕ_2 between the viewing direction and the planes whose projections on the image plane are separated by the 2D edge line. Table 4.5 shows the standard deviation of the error in the estimate of z_0 in the same conditions.

The results were obtained in 100 simulations for each pair of values of ϕ_1 and ϕ_2 using $N_b = N_s = 18$. As each pair of angles ϕ_1 and ϕ_2 corresponds to an infinite number of 3D step edge lines and 2D step edge lines, in each realisation the position of these lines was determined randomly.

In all simulations we used a step of *5 mm* at the edge point with coordinates x_0 and y_0. The results also include the cases in which the two 3D step edge lines corresponding to the 2D step edge line are not parallel.

In all the simulations used to build Tables 4.5 and 4.6 the maximum absolute value of

the error in the estimate of y_o, as expected, was always approximately equal to *0.25 mm* (equal to half of the pixel size).

As we can see from the tables, the smaller errors occurred to the smallest values of ϕ_1 and ϕ_2, *i.e.* in the cases in which the height of the plane does not vary much with the coordinates x and y. This result was expected as the bigger the slopes of the planes, the bigger the effect of the error in the estimate of y_o and z_0.

It is interesting to compare Tables 4.2 and 4.4 with Tables 4.6 and 4.5. The comparison shows a clear reduction in the accuracy in the estimate of z_o in the case of the step edges. This reduction is due to the C_0 discontinuity that does not allow the use of subpixel accuracy in the determination of the edge points.

To verify the variation of the errors of the estimate with the number of pixels used in the extraction ($N_{pixel} = N_b = N_s$) we proceeded with simulations in which we verified the variation of the error of the estimate with N_{pixel} for a given viewing condition corresponding to the same situation of Figure 4.5 and considered a step discontinuity of *5 mm* in such a way that the two 3D step edge lines were parallel

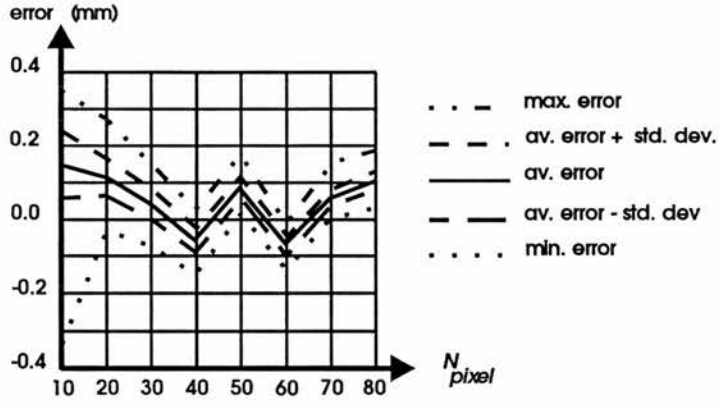


Figure 4.16: Maximum error, standard deviation, average error, minus standard deviation and minimum error of estimate of z_o. Result of 300 realisations to each value of N_{pixel} under the same conditions of Figure 4.5. Step of *5 mm* between 3D edge lines at edge point extracted.

As one could expect, for this fixed position of the 2D edge line on the image plane, the error of the estimate of y_o was constant and with an absolute value smaller than half of the pixel size. The graphs on Figure 4.16 show the maximum error, the average error plus one standard deviation, the average error, the average error minus one standard deviation and the minimum error in the estimate of z_o. The graphs show that the

estimate of z_o is clearly biased and the error of the estimate presents a constant and a random component as expected.

An interesting difference between the graphs on Figure 4.16 and 4.6 is the fact the that we do not observe a considerable reduction in the errors when N_{pixel} increases for N_{pixel} bigger than 20. This happens because the error in the estimate of y_o does not decrease when N_{pixel} increases.

The bias in the error of the estimate of the 3D edge points makes the exact determination of the uncertainty zones more complicated, as the value of the bias error may vary with the relative position of the 2D edge line and the range sensor.

Because the algorithm determines the y (or x) and z coordinates of the 3D edge point associated with a given x (or y) coordinate, the uncertainty region of the algorithm corresponds to a planar region, *i.e.* a subset of the plane yz (or xz).

If we consider the estimated values $\hat{y_o}$ (or $\hat{x_o}$) and $\hat{z_o}$ of the y (or x) and z coordinates of a 3D edge point, from all that was discussed so far the actual coordinates of the 3D edge point should be inside a rectangular region centered at $\hat{y_o}$ ($\hat{x_o}$) and $\hat{z_o}$ such that if a point with coordinates y (x) and z belong to this region, then:

$$\|y - \hat{y_0}\| \leq \frac{\Delta}{2} \tag{4.3.20}$$

$$\|z - \hat{z_0}\| \leq z_{cte} + z_{random} \tag{4.3.21}$$

where:

- Δ is the size of the pixel in the y (or x) direction
- z_{random} is the absolute value of the random component of the error in the estimate of z_o
- z_{cte} is the absolute value of the constant component of the error of the estimate of z_o

The random component of the error in the estimate of z_o in Equation (4.3.21) is similar to the random component of the error in the estimate of z_o in the case of roof edges.

We shall model this component of the error as a normally distributed random variable with mean zero and standard deviation equal to σ_z. Therefore, the probability P of z_{random} being bigger than a given positive value z_{LIM} is given by:

$$P = 1 - \frac{1}{\sigma_z\sqrt{2\pi}} \int_{-z_{LIM}}^{z_{LIM}} e^{-\frac{1}{2}(\frac{x}{\sigma_z})^2} dx \tag{4.3.22}$$

We cannot know exactly the value of z_{cte}, however we can we can estimate its maximum value (z_{cte}^{max}) using the value of k_1^* (k_2^* if we are interested in the other 3D edge line) obtained in the evaluation of $f_{distance}(\hat{y_0})$:

$$z_{cte}^{max} = \|k_1^*\| \frac{\Delta}{2} \tag{4.3.23}$$

Considering Equations (4.3.21) to (4.3.23) we can associate with the estimated coordinates of the 3D edge point ($\hat{y_o}$ and $\hat{z_o}$) a rectangular uncertainty region, defined by Equations (4.3.24) and (4.3.25), such that the probability of the actual values of the step edge coordinates being outside the uncertainty region is smaller than P (given by Equation (4.3.22)).

$$\|y - \hat{y_0}\| \leq \frac{\Delta}{2} \tag{4.3.24}$$

$$\|z - \hat{z_0}\| \leq z_{cte}^{max} + z_{LIM} \tag{4.3.25}$$

The determination of the uncertainty region defined by Equations (4.3.24) and (4.3.25) assumes the modelling of the random errors in the estimate of z_0 as a normally distributed random variable with zero mean and the accurate determination of σ_z.

Table 4.5 shows how the value of σ_z can vary considerably with the viewing conditions. Although the Table 4.5 can be used for obtaining a rough estimate of σ_z, the accurate determination of σ_z, under a given set of conditions, demands the realisation of simulations under the same conditions.

Figure 4.17 shows the actual value of probability P as a function of z_{LIM} obtained as a result of 300 simulations under the same conditions used in Figure 4.16 for $N_{pixel} = 20$. Figure 4.17 also shows the graph of two different normal estimates of P calculated using Equation (4.3.22):

- The first normal estimate is a global one, in which the value of σ_z was calculated considering the results of the 300 simulations using an equation similar to Equation 4.3.6.

- The second normal estimate is a local estimate whose objective is to estimate P for values of P around 0.02. The local estimate was obtained by scaling the estimate of σ_z in such a way that the graph of the actual estimate and the graph of the local approximation passed by the same point when $P = 0.02$.

As illustrated by the graphs of Figure 4.17, the random error in the estimate of z_o follows well our normal model and the use of local normal approximations makes it possible to obtain a very accurate normal approximation to the probability P for a chosen interval of values of P.

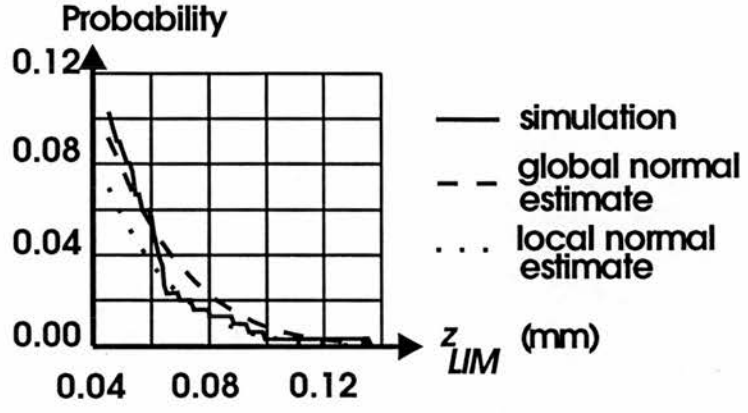


Figure 4.17: Graphs of probability P of absolute value of random error being bigger than z_{LIM}. Graph of P obtained through simulation, global normal approximation of P and local approximation of P calculated to minimise error for $P = 0.02$. Result of 300 simulations under the same conditions of Figure 4.16 with $N_{pixel} = 20$.

4.3.2.4 General properties of algorithm

From all that it was explained in the previous section, there are a few general characteristics of the algorithm we may announce:

1. The algorithm has the drawback of being biased, *i.e.* for a given position of the 2D edge line on the image plane the error in the estimate of y_o (or x_o) is constant and not necessarily equal to zero.

2. The absolute value of the error in the estimate of y_o (or x_o) in most of the cases is at most equal to half of the size of the range image pixel size in the y (or x) direction.

3. The error in the estimate of z_o is always biased and the errors of the estimate increases with the slope of the planes whose projection on the image plane have the 2D edge line as the border.

4. The error in the estimate of z_o has a constant component (due to the constant error in the estimate of y_o (or x_o) and a random component (due mainly to the noise in the range image).

As a final comment on the errors in the estimate of the 3D edge points, we should say that the performance of the algorithm can change according to the size of the C_0 discontinuity corresponding to the 3D edge point being extracted. This happens because when the discontinuity is too small the determination of the minimum of function $f_{distance}$ becomes less accurate, as it is illustrated by Figure 4.18.

This figure shows the function $f_{distance}$ corresponding to a simulation in the same conditions as Figure 4.15 but with a discontinuity of size *0.3 mm.*

As we can see in the figure, we might have errors equal to the size of the range image pixel in the y direction (*0.5 mm*) in the estimate of y_o (due to the fact that the step at the bottom of $f_{distance}$ is wider than in Figure 4.15).

In our simulations we observed that as long as the size of the *step* in the parametric model of Equation (4.3.14) is bigger than a given minimum value, the performance of the algorithm does not change significantly with the size of the step.

According to our simulations, for laser stripers with $\sigma_S = 0.1$ *mm* the minimum value of *step* is equal to approximately *3 mm* and to laser striper with $\sigma_S = 0.01$ *mm* the minimum value of *step* is equal to approximately *0.8 mm.*

4.3.3 Extraction of step edges corresponding to holes and determination of hole's center and hole's radius

4.3.3.1 Objectives and motivation

The objective of this section is the determination of the 3D step edge points created by the intersection of a plane and a circular cylinder when the viewing direction is parallel to the plane normal and to the axis of the cylinder.

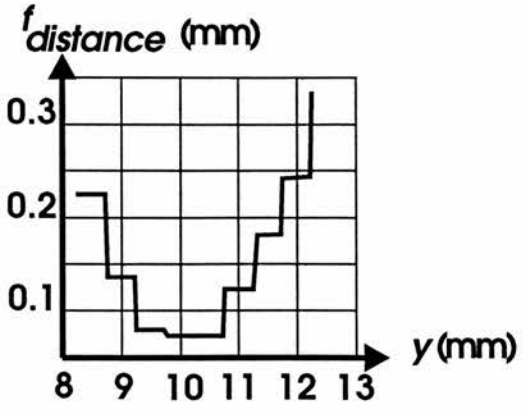


Figure 4.18: Function $f_{distance}$ obtained in the extraction of step edge. Result of simulation under the same conditions of Figure 4.15, but with a C_0 discontinuity equal to *0.3 mm* in the step edge.

Our final objective is the inspection of circular cylindrical holes in manufactured objects. The restrictions in the viewing conditions of the hole were created in order to maximise the accuracy in the inspection of the hole's border (circle corresponding to the projection of the cylinder on the image plane) which is the geometrical element we want to inspect. We will also only consider holes corresponding to C_0 discontinuities of at least *3 mm*.

4.3.3.2 Algorithm description

We determine the edge points of the upper border of the cylindrical hole using the same algorithm of Section 4.3.2 (notice that we are only interested in the projections of the edge points on the image plane). The extraction of the edge points is done after the estimation of the radius and the center of the hole border. These estimates are used to obtain an initial approximation of the hole border that is used during the determination of the edge points.

The initial estimates of the radius and the hole's center are calculated using an approximation of the region in the image plane belonging to the inside of the hole's border (that we will call inner region). The center of mass of the inner region is used to estimate the center of the hole. The radius of the hole is estimated by calculating the maximum distance between a point belonging to the inner region and the estimated center.

The inner region of the hole is determined by thresholding the z coordinate of the range image. The thresholding value is chosen taking into account the average value of the z coordinate in points outside the hole border (z_{AV}) and the standard deviation of the sensor noise (σ_S). We use the threshold value equal to $z_{AV} - 4\sigma_S$.

After the extraction of the edge points of the hole's border, the edge points are used to calculate a new and more accurate estimate of the radius of the hole's border. The new estimate is calculated in such a way that the average distance between the edge points and the approximation of the hole's border is minimum.

It is interesting to observe that we do not use the edge points to estimate the center of the hole border, as the errors of the estimate calculated using the inner region are smaller. This happens because the extracted edge points in general will present a bigger asymmetry with relation to the center of the hole than the inner region. The value of the radius estimate ($\hat{radius}$) is calculated according to Equation (4.3.26).

$$\hat{radius} = \sqrt{\frac{1}{N_{feat}} \sum_{F_m} (x_i - \hat{x_{center}})^2 + (y_i - \hat{y_{center}})^2} \tag{4.3.26}$$

where:

- N_{feat} is the number of extracted edge points in the set of measured features F_m,
- x_i and y_i are respectively the x and y coordinates of the measured edge points and
- $\hat{x_{center}}$ and $\hat{y_{center}}$ are the estimates of the hole's center

4.3.3.3 Error analysis

The errors in the extraction of the edge step points corresponding to the hole's border follows basically the same behaviour described in the previous section. As before, the errors are biased and the values of the bias vary according to:

- the position of the center of the hole with relation to the pixels of the range image,
- the circle radius,

- the relative position of the projection of the edge point on the hole border.

An important difference between this case and the previous section is the fact that we have errors in both coordinates (x and y) of the edge points on the image plane. This happens because the errors now correspond to the distance between the extracted edge point and the closest point on the actual hole's border.

Figure 4.19 illustrates the errors in the x and y coordinates of the edge points corresponding to a hole with radius *10 mm* and C_0 discontinuity of *5 mm* when the center of the hole's border is in the middle of a pixel.

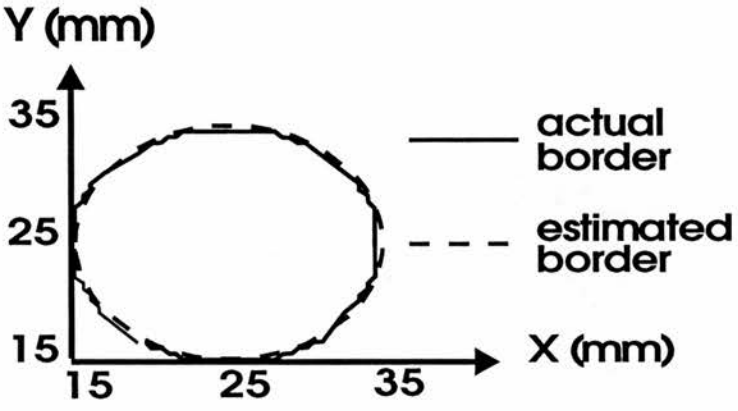


Figure 4.19: Actual and estimate hole border. Result of simulation for a hole with radius *10 mm*, C_0 discontinuity equal to *5 mm*. Simulation of sensor with pixels of *0.5 mm* × *0.5 mm* and standard deviation σ_S = *0.1 mm*. Center of hole border in the middle of pixel.

Another important difference between the errors in this case and in Section 4.3.2 is the fact that we might have errors bigger than half of the pixel size. This happens because the projection of the C_0 discontinuity on the image plane is now a circle and, as a consequence of this fact, the function $f_{distance}$ becomes similar to the one in Figure 4.18. Accurate estimates of the maximum error in the extraction of edge points can be obtained by simulations.

The errors on the final estimate of the radius of the hole's border are shown in Figure 4.20. This figure shows the variation of the maximum absolute value of the error in the radius estimate as a function of the actual value of the hole's radius. The figure was obtained as a result of 100 simulations to each value of radius. The position of the center of the hole's border was generated randomly in each simulation.

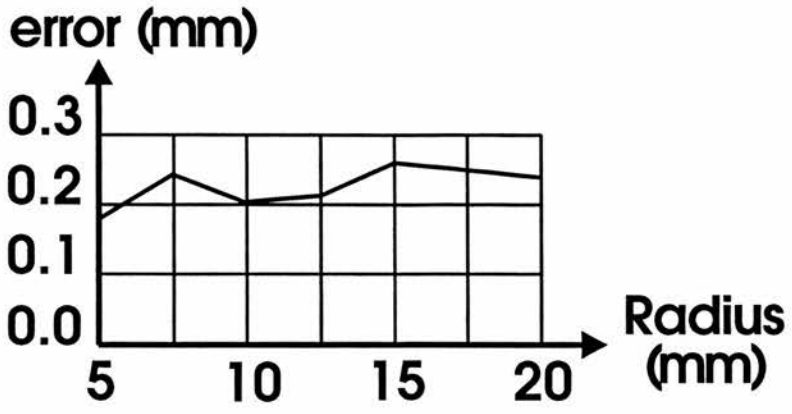


Figure 4.20: Maximum absolute error in the radius estimate as a function of the actual radius size. Result of 100 simulations for each radius with a C_0 discontinuity equal to *5 mm*. Simulation of sensor with pixels of *0.5 mm* × *0.5 mm* and standard deviation $\sigma_S = 0.1$ *mm*. Center of hole border randomly generated in each simulation.

Figure 4.21 shows the maximum absolute error in the estimate of the center of the hole's border under the same simulation conditions as Figure 4.20 as a function of the radius of the actual border of the hole.

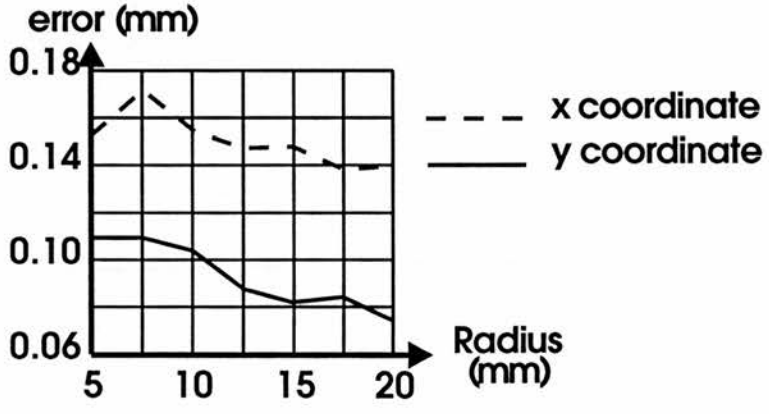


Figure 4.21: Maximum absolute error in the estimate of the coordinates of the center of the hole's border as a function of the actual radius size. Result of 100 simulations for each radius for a hole with a C_0 discontinuity equal to *5 mm*. Simulation of sensor with pixels of *0.5 mm* × *0.5 mm* and standard deviation $\sigma_S = 0.1$ *mm*. Center of hole border randomly generated in each simulation.

An important property of the estimates of the radius and the center of the hole's border is the fact that these estimates are biased, *i.e.* for a given position of the hole's center and a given size of the hole's radius, the expected error of the estimates is not zero. This fact makes the modelling of the uncertainty regions of the estimates much more complicated. We will use Figures 4.20 and 4.21 to build a simple model of the uncertainty regions.

In order to do so, we will assume that the size of the uncertainty region is equal to the values of the maximum error of the estimates observed in the Figures 4.20 and 4.21. Besides that, we will also assume that the probability of the error being bigger than the maximum value observed in the simulation is equal to $\frac{1}{N_{total}}$ (where N_{total} is the number of simulations considered). This approximation is not strictly correct, and is justified by the idea that the confidence in the estimate of the size of the uncertainty region increases when the number of simulations increases.

In the case of the center of the hole's border we will use this idea to determine the maximum absolute values of the errors in the estimates of the x coordinate (X_{max}) and the y coordinate (Y_{max}). After that, we will approximate the uncertainty region by a disc of radius R_{lim} (centered at the measured value of the holes's center) given by:

$$R_{lim} = \sqrt{{X_{max}}^2 + {Y_{max}}^2} \tag{4.3.27}$$

4.3.3.4 General properties of algorithm

As a final comment on the errors of the estimates of the hole's radius and the hole's center, we should say that:

- the size of the uncertainty regions do not vary much with the variation of σ_S. For instance, for a reduction of the value of σ_S from *0.1 mm* to *0.01 mm* we observed a reduction in the size of the uncertainty region of only approximately 20 %,
- the accurate determination of the edge points demands the use of at least 10 pixels in the inner region of the hole. Because of this fact the method described can only be used with holes with radius of at least approximately *8 pixels* (*4 mm* if the pixel is of size *0.5 mm* $\times$ *0.5 mm*).

4.3.4 Extraction of Vertices

4.3.4.1 Objectives and motivation

In this section we will be concerned with the determination of vertices (3D points) corresponding to the intersection of at least 3 planes. The reason for our concern with

the determination of vertices is our intention to use the extracted vertices to inspect the size (length) of segments of a line resulting from the intersection of two planes.

4.3.4.2 Algorithm description

Let us start by considering the problem of determining the 3D coordinates of the vertex A of the segment AB shown in Figure 4.22. In order to estimate the vertex we will calculate the intersection between the segment AB and another segment (segment AC in Figure 4.22) passing by the vertex A.

Without loss of generality we will assume that both segments are visible in a unique range image, and also that AB corresponds to a roof edge and AC to a step or roof edge. The determination of A will be carried out according to the following procedure:

1. Using the algorithms for extraction of 3D edge points belonging to lines described in Sections 4.3.1 and 4.3.2 determine a set of 3D edge points belonging to the interior of the segment AB. We shall call this set of 3D edge points $Point_{AB}$.
2. Calculate the least square approximation of the best line corresponding to the set of points $Point_{AB}$ (that we will call $Line_{AB}$).
3. Repeat the two previous items for the segment AC obtaining the set of points $Point_{AC}$ and the least square approximation $Line_{AC}$
4. Use the intersection between the projections of $Line_{AB}$ and $Line_{AC}$ on the image plane to determine the estimates $\hat{x_A}$ and $\hat{y_A}$ of the coordinates of the vertex A.
5. Determine the estimate $\hat{z_A}$ of the z coordinate of vertex A using $\hat{x_A}$, $\hat{y_A}$ and $Line_{AB}$.

The accuracy in the determination of A using the procedure just described depends on 3 factors:

- the accuracy in the extraction of the 3D edge points (the more accurate the extraction the better)
- the number of 3D edge points in P_{AB} and P_{AC} (the more points used the better)

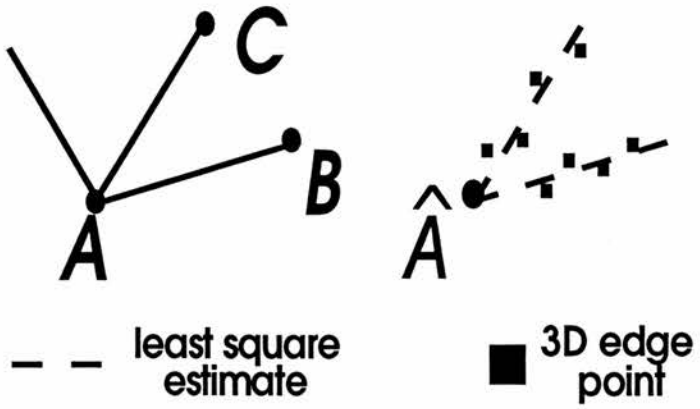


Figure 4.22: Determination of vertex corresponding to the intersection of two segments of line.

- the angle between the projection of segments AB and BC on the image plane (the nearer the angle to 90° the better)

As in the previous sections, the accuracy of the extraction algorithm increases when the algorithm become less local. This happens because when we reduce the neighbourhood of A used in the extraction we also reduce the number of 3D edge points and the accuracy in the extraction of the 3D edge points.

Let us assume that projections of $Line_{AB}$ and $Line_{AC}$ on the image plane are described, according to Equation (4.3.28), by the constants a_{AB}, b_{AB} and c_{AB} and a_{AC}, b_{AC} and c_{AC} respectively. In this case, the estimates $\hat{x_A}$ and $\hat{y_A}$ can be calculated using Equations (4.3.29) and (4.3.30).

$$ax + by + c = 0 \tag{4.3.28}$$

$$\hat{x_A} = \frac{b_{AC}c_{AB} - b_{AB}c_{AC}}{a_{AB}b_{AC} - a_{AC}b_{AB}} \tag{4.3.29}$$

$$\hat{y_A} = \frac{a_{AB}c_{AC} - a_{AC}c_{AB}}{a_{AB}b_{AC} - a_{AC}b_{AB}} \tag{4.3.30}$$

It is important to observe that because the segments AB and AC will never be parallel we will always be able to use Equations (4.3.29) and (4.3.30). However, the more parallel are the segments the more sensitive will be the estimates of $\hat{x_A}$ and $\hat{y_A}$ to errors in the estimates of the constants defining the segments AB and AC.

This happens because the first derivatives of $\hat{x_A}$ and $\hat{y_A}$ with the respect to the constants defining the segments AB and AC approach infinity when the two segments be-

come nearer and nearer to parallel segments (K_{angle} approaches 0 in Equation (4.3.32)). This fact is illustrated by Equation (4.3.32) that shows the first derivative of $\hat{x_A}$ with respect to a_{AC}.

$$K_{angle} = a_{AB}b_{AC} - a_{AC}b_{AB} \tag{4.3.31}$$

$$\frac{\delta \hat{x_A}}{\delta a_{AC}} = \frac{b_{AB}(c_{AB}b_{AC} - c_{AC}b_{AB})}{{K_{angle}}^2} \tag{4.3.32}$$

The estimate of z_A ($\hat{z_A}$) is calculated using either the projection of the $Line_{AB}$ on the plane xz and $\hat{x_A}$ or the projection of the $Line_{AB}$ on the plane yz and $\hat{y_A}$. The choice of the estimate to be used in the calculation is based on the uncertainty regions of the 3D points extracted from AB. The estimate $\hat{x_A}$ is used when the uncertainty regions are subsets of the plane yz and the estimate $\hat{y_A}$ is used otherwise.

In order to model the error in the estimate of the coordinates of the vertex A we proceeded with Monte Carlo simulations in which we observed the variation of the estimate errors with the number of 3D edge points in $Point_{AB}$ and $Point_{AC}$, the angle α between the projections of AB and AC on the image plane and the accuracy of the 3D edge points.

Taking into account the details of the extraction of the 3D edge points described in Section 4.3.1 and Section 4.3.2, during the simulations we considered errors in only two of the three coordinates of the 3D edge points (x and z or y and z). The error in the coordinate of the 3D edge point on the image plane:

- is therefore modeled as a normally distributed random variable with mean zero and standard deviation σ_{edge} in the case of roof edges or
- arises from the difference between the projection of the actual edge point on the image plane and the projection of the edge point that would be extracted by the step edge detector described previously.

The error in the estimate of the z coordinate of the 3D edge points corresponding to roof edges was assumed as normally distributed with mean equal to zero and standard deviation σ^z_{edge}.

In all the simulations the estimate of the constants a, b and c defining the projections of

$Line_{AB}$ and $Line_{AC}$ on the image plane were calculated according to either Equations (4.3.34) to (4.3.36) or Equations (4.3.37) to (4.3.39). Equations (4.3.34) to (4.3.36) were used when the estimated angle between the projection of the segment on the image plane and the x direction was smaller than 45° and Equation (4.3.37) to (4.3.39) in the other cases.

These equations determine the values a^*, b^* and c^* that minimise the average distance d_{AV} (see Equation (4.3.33)) between the projection of the 3D edge points on the image plane and the least square approximation. Equations (4.3.34) to (4.3.36) were also used, with the z coordinate in the place of the y (or x) coordinate, to determine the relation between the x (or y) and the z in $Line_{AB}$.

$$d_{AV} = \frac{1}{N_{points}} \sum_{P_{edge}} (ax_i + by_i + c)^2 \tag{4.3.33}$$

$$a^* = \frac{\sum_{P_{edge}} x_i \sum_{P_{edge}} y_i - N_{points} \sum x_i y_i}{N_{points} \sum_{P_{edge}} {x_i}^2 - \sum_{P_{edge}} {x_i}^2} \tag{4.3.34}$$

$$b^* = 1 \tag{4.3.35}$$

$$c^* = \frac{\sum_{P_{edge}} x_i y_i \sum_{P_{edge}} x_i - \sum_{P_{edge}} {x_i}^2 \sum_{P_{edge}} y_i}{\sum_{P_{edge}} {x_i}^2 - \sum_{P_{edge}} {x_i}^2} \tag{4.3.36}$$

$$a^* = 1 \tag{4.3.37}$$

$$b^* = \frac{\sum_{P_{edge}} x_i \sum_{P_{edge}} y_i - N_{points} \sum x_i y_i}{N_{points} \sum_{P_{edge}} {y_i}^2 - \sum_{P_{edge}} {y_i}^2} \tag{4.3.38}$$

$$c^* = \frac{\sum_{P_{edge}} x_i y_i \sum_{P_{edge}} y_i - \sum_{P_{edge}} {y_i}^2 \sum_{P_{edge}} x_i}{\sum_{P_{edge}} {y_i}^2 - \sum_{P_{edge}} {y_i}^2} \tag{4.3.39}$$

where:

- P_{edge} is the set of 3D edge points
- x_i and y_i are the coordinates of ith 3D edge point in P_{edge}

4.3.4.3 Error analysis

Figures 4.23 and 4.24 show the variation of the error of the estimates $\hat{x_A}$ as a function of the angle between the projection of segments AB and AC on the image plane. The figures are the result of 100 simulations for each angle considered when the segment

AB makes an angle of *0.7 rad* with the x axis and the image plane. Both figures were obtained considering 10 edge points in $Point_{AC}$ and 30 edge points in $Point_{AB}$.

Figure 4.24 corresponds to the case in which AC is a step edge and the maximum error in the estimate of the projection of a 3D step edge point on the image plane is *0.25 mm*. Figure 4.23 corresponds to the case in which AC is a roof edge. In both figures, during the simulation of the errors in the extraction of 3D roof edge points, it was assumed that $\sigma_{edge} = \sigma^z_{edge} = 0.1\ mm$.

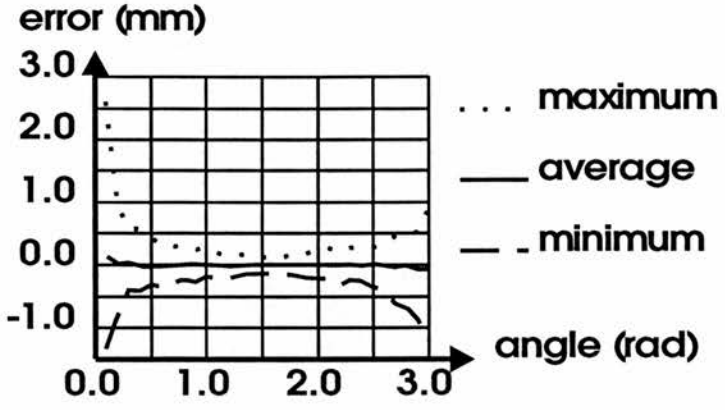


Figure 4.23: Error of the estimate $\hat{x_A}$ as a function of the angle between the projections of the segments AB and AC on the image plane. Result of 100 simulations for each angle between the segments when: AC and AB are roof edges, the projection of AB on the image plane makes an angle of *0.7 rad* with the axis x and AB makes an angle of *0.7 rad* with the image plane. $Point_{AB}$ with 30 pixels and $Point_{AC}$ with 10 pixels. Simulation of laser striper with pixels of size $0.5\ mm \times 0.5\ mm$ and viewing conditions such that $\sigma_{edge} = \sigma^z_{edge} = 0.1\ mm$.

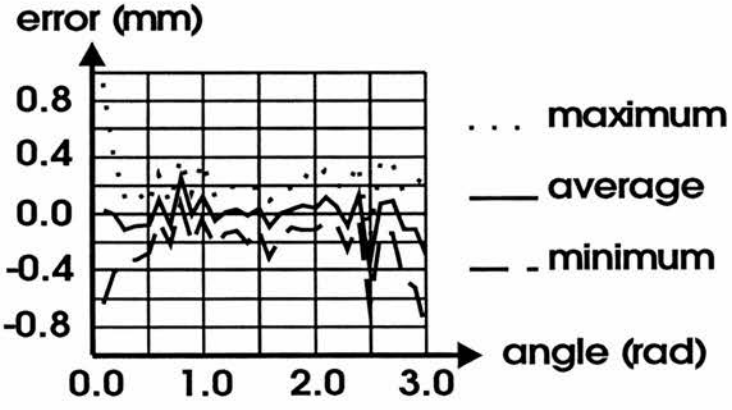


Figure 4.24: Error of the estimate $\hat{x_A}$ as a function of the angle between the projections of the segments AB and AC on the image plane. Result of 100 simulations for each angle between the segments when: AC is a step edge, AB is a roof edge. Simulation under the same condition of Figure 4.23.

Although the graphs on Figures 4.23 and 4.24 show the errors just on the x coordinate

of the vertex A they illustrate well the error in the estimate of the vertex A, as the graphs of the errors on the estimate of the y and z are very similar and present errors on the same order of magnitude.

The graphs also illustrate our previous comments on the variation of the error with the angle between the segments AB and AC. As expected we observed a great degradation in performance when the two segments are nearly parallel (with errors of the order of *1 mm*). Concerning the difference in performance between the two figures we can observe that:

- In the case of Figure 4.23 the estimation of the vertex A did not present any bias, contrary to Figure 4.24 in which the estimator is clearly biased. This happened because the estimate of the 3D edge points in the case of Figure 4.24 is also biased.

- In both cases the magnitude of the errors in the estimate was about the same for angles near *1.57 rad* and the errors in the case of Figure 4.24 were clearly smaller for angles near *0 rad.*

  This happened because the average error of the extraction of the 3D edge points in both cases was about the same, but the maximum possible error in the case of Figure 4.23 was bigger. Therefore, for angles in which the determination of the vertex A is more sensitive to the error of a single 3D edge point we observed bigger errors in the case of Figure 4.23.

Despite being obtained under particular simulation conditions, the graphs on Figures 4.23 and 4.24 illustrate the general qualitative behaviour of the errors in the estimate of the coordinates of the vertex A with the variation of the angle α.

The variation in the performance of the algorithm with the number of 3D edge points in $Point_{AB}$ is illustrated in Figure 4.25 that shows the maximum, the average and the minimum error in the estimate of z_A. The figure was obtained as the result of 200 simulations to each value of the number of pixels in $Point_{AB}$ under the same simulation conditions of Figure 4.23 and for $\alpha = 1.5\ rad$.

The figure shows clearly the reduction of the maximum errors in the extraction when

we increase the number of 3D edge points. The graphs of the errors in the estimate of x_A and y_A are similar to the graphs on Figure 4.25 and present errors with the same order of magnitude.

The results shown in Figure 4.25 illustrate also the variation of the error in the estimates under the same simulation conditions of Figure 4.24. The difference being, as we discussed before, the fact that the errors are biased in this case.

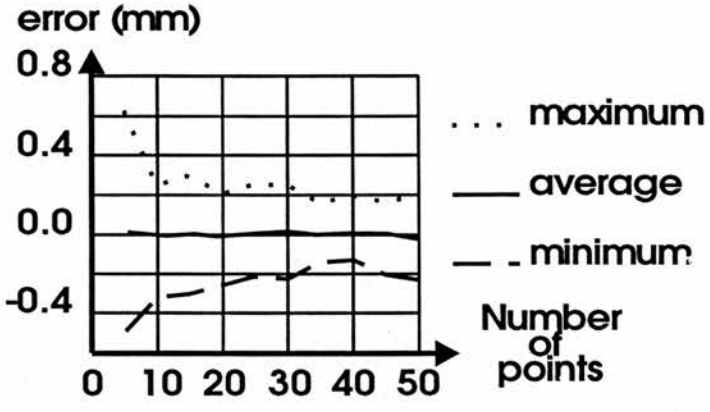


Figure 4.25: Error of the estimate $\hat{z_A}$ as a function of the number of 3D edge points in $Point_{AB}$. Result of 200 simulations to each number of 3D edge points in $Point_{AB}$ when: AC and AB are roof edges. Simulation under the same conditions of Figure 4.23 using 10 3D edge points from segment AC.

As one should expect, the variation of the errors of the estimates with the number of 3D points in $Point_{AC}$ is similar to the one shown in Figure 4.25 when AC is a roof edge. However, when AC is a step edge the main effect of varying the number of 3D edge points in $Point_{AC}$ is the variation of the bias error in the estimate of the vertices coordinates.

This fact is illustrated by Figure 4.26 that shows the variation of the average errors observed under the same simulation conditions of Figure 4.24 for $\alpha = 1.5$ *rad.* The figure was obtained as the result of 200 simulations for each number of 3D edge points in $Point_{AC}$.

From now on we will assume that the estimate of the vertex position is always unbiased, *i.e.* the expected error of the estimate is negligible (one or two orders of magnitude smaller than the standard deviation of the error of the estimate).

We will approximate the uncertainty regions associated with the estimate of the vertex

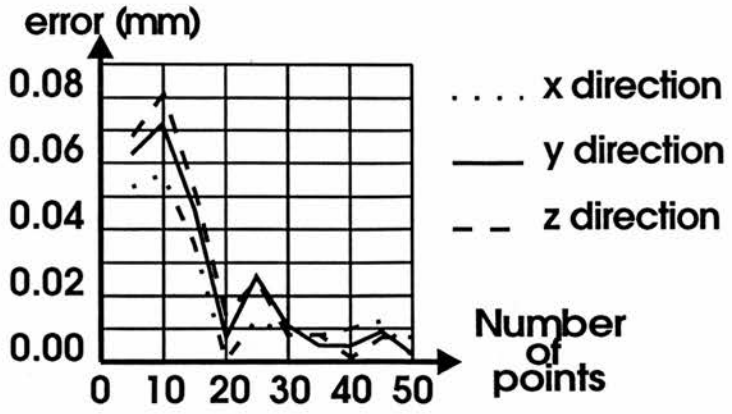


Figure 4.26: Average error of the estimates of the vertex coordinates as a function of the number of 3D edge points in $Point_{AC}$. Result of 200 simulations for each number of 3D edge points in $Point_{AC}$ when AC is a step edge. Simulation under the same conditions as Figure 4.23 using 30 3D edge points from segment AB.

A using ellipsoids. This approximation is equivalent to modelling the errors of the estimate as a normally distributed random vector with mean equal to zero.

As in the previous sections, we will determine the variance of the errors (that we will call Var_A) using the result of Monte Carlo simulations (the elements of matrix Var_A are determined applying equations similar to Equation (4.3.6) to the result of the simulations).

The main axis of the ellipsoids used to approximate the uncertainty regions (that we will call $Axis_1$, $Axis_2$ and $Axis_3$) can be determined considering the probability P_A of the actual vertex being outside the uncertainty region and the eigenvalues of the variance matrix Var_A that we call λ_1, λ_2 and λ_3.

Let us consider an ellipsoid such that:

- the ellipsoid is centered in $\hat{x_A}$, $\hat{y_A}$ and $\hat{z_A}$ and
- the main axes of the ellipsoid are parallel to the eigenvectors of Var_A and
- the dimension of the main axis of the ellipsoid are given by:

$$Axis_1 = K_0\lambda_1 \tag{4.3.40}$$

$$Axis_2 = K_0\lambda_2 \tag{4.3.41}$$

$$Axis_3 = K_0\lambda_3 \tag{4.3.42}$$

where: K_0 is a constant that determines the size of the ellipsoid.

If we define an uncertainty region limited by the ellipsoid just described, the normal estimate of the probability of the actual values of x_A, y_A and z_A being outside the uncertainty region (that we will call $\hat{P_A}$) is given by Equation (4.3.43).

$$\hat{P_A} = 1.0 - \frac{2}{\sqrt{2\pi}} \int_0^{K_0} x^2 e^{\frac{-x^2}{2}} dx \tag{4.3.43}$$

Figure 4.27 shows the variation of the actual value of P_A (obtained in the simulation) and its normal estimate $\hat{P_A}$ with the value of K_0. The result was obtained as a result of 1000 simulations under the same conditions of Figure 4.23 for α =*1.5*. The graphs illustrate how $\hat{P_A}$ underestimates the actual value of P_A.

To correct this error, as in Section 4.3.1, we will scale the main axis of the ellipsoid defining the uncertainty region by a factor K_{scale} and consider a new normal estimate of P_A given by Equation (4.3.44).

The scaling aims to determine an statistical approximation of the errors that makes the difference between P_A and $\hat{P_A}$ equal to zero for a desired value of P_A.

Figure 4.28 shows the actual value of P_A obtained in the same conditions of Figure 4.27 and its normal estimate (calculated to make $\hat{P_A} = P_A$ for $P_A = 0.01$).

$$\hat{P_A} = 1.0 - \frac{2}{\sqrt{2\pi}} \int_0^{\frac{K_0}{K_{scale}}} x^2 e^{\frac{-x^2}{2}} dx \tag{4.3.44}$$

As before, we will use Monte Carlo simulations for the construction of a statistical model of the error in the measurement of the length of the segment AB under a particular set of conditions and we will try to approximate the error by a normally distributed variable.

Figure 4.29 illustrates the typical variation of the errors in the estimate of the length of the segment AB as a function of the number of edge points considered in segment AB. The graph in the figure was obtained as the result of 200 simulations to each number of edge points in which we:

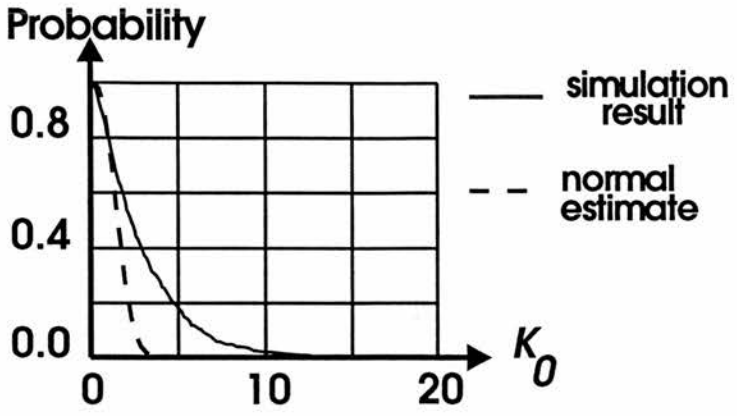


Figure 4.27: Probability P_A of measurement being outside uncertainty region and its approximation using a normally distributed random variable. Result of 1000 simulations when α *=1.5 rad*, AC and AB are roof edges. Simulation under the same conditions as Figure 4.23 using 30 3D edge points from segment AB and 10 3D edge points from segment AC.

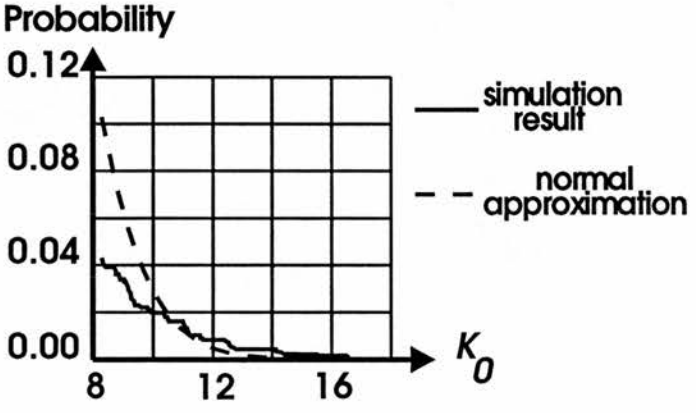


Figure 4.28: Probability P_A of measurement being outside uncertainty region and its approximation using a normally distributed random variable built to reduce the errors for $P_A = 0.01$. Result of 1000 simulations under the same conditions of Figure 4.23.

- considered only roof edges,
- determined the vertices using 20 edge points in all the auxiliary segments used to determine the vertices A and B, and
- consider an angle of *1 rad* between AB and the auxiliary segments used in the extraction of the vertices.

The graph of Figure 4.29 can be used for obtaining a rough estimate of the size of the uncertainty region (that in this case is a segment) in the measurement of the length of the segment AB as a function of the number of edge points. However, a more accurate

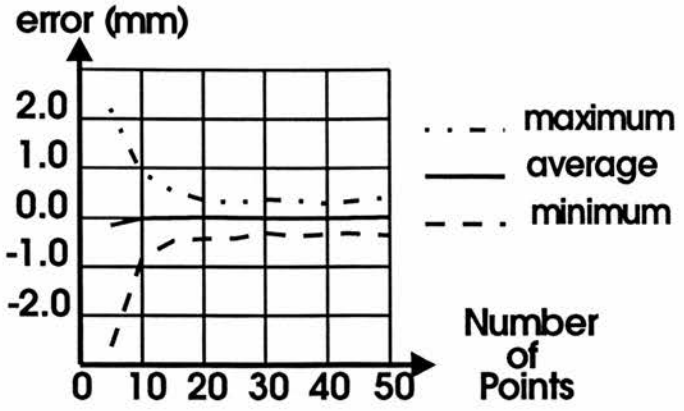


Figure 4.29: Maximum error, average error, and minimum error of the estimate of the length of segment AB as a function of the number of edge points in AB. Result of 200 simulations to each number of 3D edge points in $Point_{AB}$ when all the auxiliary segments used in the extraction are roof edges with 20 edge points, AB is a roof edge, the projection of AB on the image plane makes an angle of *0.7 rad* with the axis x and AB makes an angle of *0.7 rad* with the image plane. Angle between auxiliary segments and segment AB equal to *1 rad*. Size of segment AB equal to *30 mm*. Simulation assuming the same sensor and errors in the extraction of the edge points considered in Figure 4.23.

determination of the uncertainty regions can be obtained by simulation, as it is shown in Figure 4.30.

Figure 4.30 shows the graph of the probability P of the absolute value of the error in the measurement of the length of the segment AB being bigger than L_{lim}. The figure was obtained as the result of 1000 simulations in the same conditions of Figure 4.29 when we consider 30 edge points in segment AB.

The figure also shows an estimate of the probability P that was built considering the errors normally distributed. The graphs shows that the actual result of the simulation follows well the normal model and that it is possible to obtain very accurate local approximations of the actual probability density function.

4.3.5 Extraction of planes

4.3.5.1 Objectives and motivation

In this section we will be concerned with the analysis of the errors made in the least square fitting of planes to range images. Our ultimate objective is to use the result of this section in the Sections 4.3.6 and 4.3.7 for:

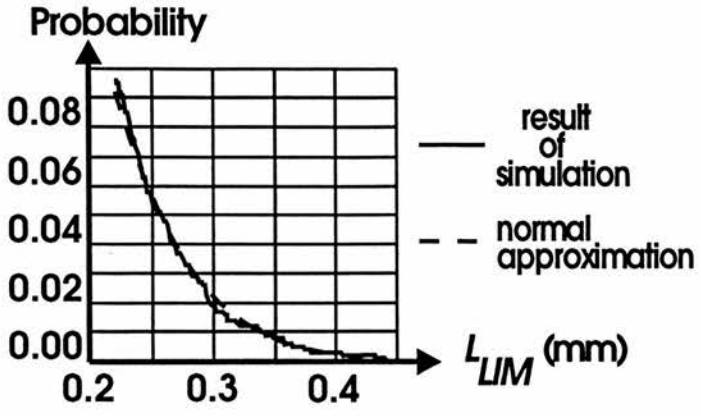


Figure 4.30: Graph of the probability P of the absolute value of the error in the measurement of length being bigger than L_{lim} and its approximation by a normally distributed random variable. Result of 1000 simulations when all the auxiliary segments used in the extraction are roof edges with 20 3D edge points and AB is a roof edge with 30 3D edge points. Simulation under the same conditions as Figure 4.29.

- to inspect the angle between two planar surfaces
- to inspect the angle between segments of line corresponding to the intersection between planar surfaces.

4.3.5.2 Algorithm description

We will assume that we have an initial estimate of the projection of the planes on the image plane (given by the registration between model and range image) that is accurate enough to allow the definition of sets of 3D points belonging to the interior of the planes whose angle we want to measure.

In order to achieve our aim, let us start by considering the error in the least square fitting of a plane given by Equation (4.3.45) (notice a plane given by $ax + by + c = 0$ would not be visible) to a set of 3D points $P_{plane} = \{p_1, p_2, ..., p_{N_p}\}$ such that the coordinates of the pixel p_i are equal to (x_i, y_i, z_i).

$$z = ax + by + c \tag{4.3.45}$$

The least square fitting corresponds to the problem of finding the estimates $\hat{a}$ of a, $\hat{b}$ of b, and $\hat{c}$ of c that minimise d_{AV} given by Equation (4.3.46). These estimates and

their variance can be calculated using Equations (4.3.47) to (4.3.52).

$$d_{AV} = \frac{1}{N_p} \sum_{P_{plane}} (z_i - ax_i - by_i - c)^2 \tag{4.3.46}$$

$$C_{lsq} = \begin{bmatrix} \sum_{P_{plane}} {x_i}^2 & \sum_{P_{plane}} x_i y_i & \sum_{P_{plane}} x_i \\ \sum_{P_{plane}} x_i y_i & \sum_{P_{plane}} {y_i}^2 & \sum_{P_{plane}} y_i \\ \sum_{P_{plane}} x_i & \sum_{P_{plane}} y_i & N_p \end{bmatrix} \tag{4.3.47}$$

$$X_{lsq} = \begin{bmatrix} \hat{a} \\ \hat{b} \\ \hat{c} \end{bmatrix} \tag{4.3.48}$$

$$B_{lsq} = \begin{bmatrix} \sum_{P_{plane}} z_i x_i \\ \sum_{P_{plane}} z_i y_i \\ \sum_{P_{plane}} z_i \end{bmatrix} \tag{4.3.49}$$

$$C_{lsq} X_{lsq} = -B_{lsq} \tag{4.3.50}$$

$$\mathbf{E}[X_{lsq}] = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \tag{4.3.51}$$

$$\mathbf{E}[(X_{lsq} - \mathbf{E}[X_{lsq}])^2] = ({C_{lsq}}^T C_{lsq})^{-1} \tag{4.3.52}$$

where: ${C_{lsq}}^T$ is the transpose of C_{lsq}.

4.3.5.3 Error analysis and general properties of the algorithm

To observe the accuracy and precision in the estimation of the constants a, b and c we proceeded with simulations in which we applied the least square fitting to synthetically created range images of planes.

These images were created by associating with each pixel p_i (with coordinates x_i and y_i in the image plane) a coordinate z_i equal to the height of an ideal plane described by Equation (4.3.45) plus a gaussian random process of standard deviation σ_S modelling the errors in the laser striper.

As a first observation of the results of the simulation tests, we must say that for N_p not smaller than 100 the average error of the estimates was always near zero which

seems to confirm the validity of Equation (4.3.51). As one should expect the errors of the estimate decreased when N_p increased (see Figure 4.32).

We observed that the variance of the estimates $\hat{a}$, $\hat{b}$ and $\hat{c}$ given by Equation (4.3.52) is not accurate for small values of N_p (smaller than 1600). Furthermore, the values given by Equation (4.3.52) approach the right values as N_p approaches infinity.

This fact is illustrated in Figure 4.31 that shows the graphs of the standard deviations of the estimates of a and b obtained in the simulation and the corresponding estimates calculated according to Equation (4.3.52). The graphs show the variation of the standard deviations as a function of the number of pixels on the side of the square region of the image plane used to calculate the estimates of a, b and c (that we will call N_{sq}).

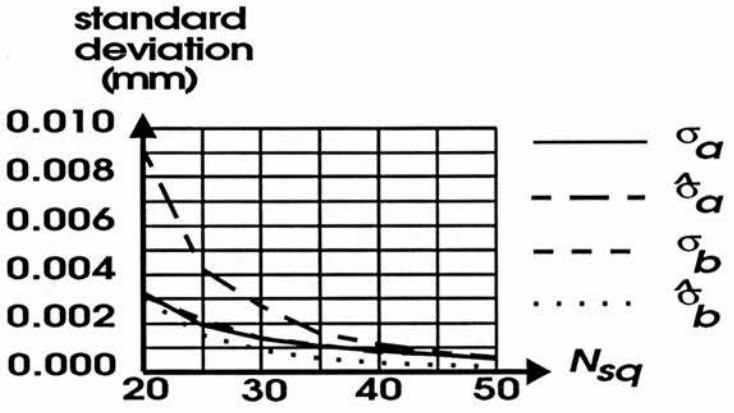


Figure 4.31: Estimate of standard deviations ($\hat{\sigma_a}$ and $\hat{\sigma_b}$) and actual standard deviations (σ_a and σ_b) as a function of N_{sq}. Result of one hundred simulations to each value of N_{sq} for $a = -1.68$, $b = 3.19$ and $c = 832.19$. Laser striper assumed to have pixels with size $0.5\ mm \times 0.5\ mm$ and $\sigma_S = 0.1\ mm$.

Although these are the results of a particular situation they illustrate a fact that occurs independent of the values of a, b and c (at least according to the results of our simulations).

A very important property of the errors of the estimates $\hat{a}$, $\hat{b}$ and $\hat{c}$ is the fact that for values of N_p bigger than or equal to 400 (N_{sq} at least equal to approximately 20) the standard deviation of the errors in the estimates of a, b and c practically do not vary with the actual values of a, b and c (this is a property to which we will not offer any proof that corresponds to a result observed in all our simulations).

Furthermore, when N_{sq} is at least equal to 20 the standard deviations of the estimates

of a and b are approximately the same (see Figure 4.31) and the covariances between the errors in the estimate of a, b and c are negligible (at least one order of magnitude smaller than $\hat{a}$, $\hat{b}$ and $\hat{c}$).

Figure 4.32 shows the graph of the maximum error, plus one standard deviation, average error, minus one standard deviation and minimum error of the estimate of c as a function of N_{sq}. The graph was obtained as the result of 100 simulations to each value of N_{sq} for $a = -1.68$, $b = 3.19$, $c = 832.19$ and σ_S equal to *0.1 mm*.

Notice that for values of N_{sq} bigger than 40 we can use Equation (4.3.52) to estimate the variance of the error of the estimate.

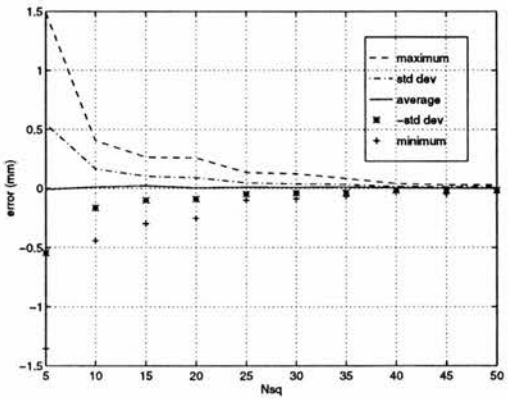


Figure 4.32: Maximum error, standard deviation, average error, minus standard deviation and minimum error of estimate of c as a function of N_{sq}. Result of one hundred simulations to each value of N_{sq} for $a = -1.68$, $b = 3.19$ and $c = 832.19$. Laser striper assumed to have pixels with size *0.5 mm* $\times$ *0.5 mm* and $\sigma_S = 0.1$ *mm*.

Another important property of the error of the estimates of a, b and c that we also observed in our simulations is the fact that the errors vary linearly with σ_S. This means, for instance, that everything else being constant the standard deviation of the errors obtained with a sensor with $\sigma_S = 0.01$ *mm* will be ten times smaller than the standard deviation of the errors obtained with a sensor with $\sigma_S = 0.1$ *mm*. This fact is illustrated by Figure 4.33 that shows the variation of the standard deviation of the error in the estimate of a as a function of the standard deviation of the error in the range measurement σ_S.

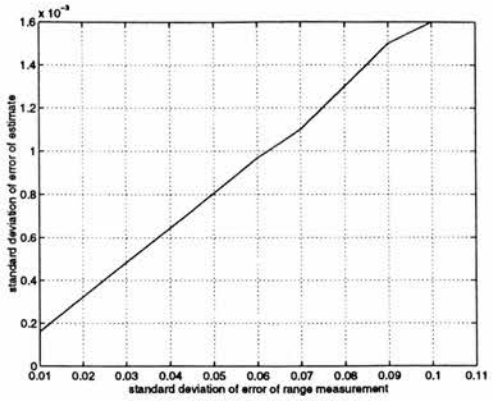


Figure 4.33: Variation of the standard deviation of the error in the estimate of a as a function of the standard deviation of the error in the range measurement σ_S. Result of 300 simulations for each value of σ_S with $N_{sq} = 25$, $a = -1.68$, $b = 3.19$ and $c = 832.19$. Laser striper assumed to have pixels with size *0.5 mm* × *0.5 mm*.

4.3.6 Extracting the angle between two planes

4.3.6.1 Objectives and motivation

Our objective in this section is to use the results of the previous section to measure the angle between two planes. We will determine the angle between two given planes using the angle between the normals of these planes. We will consider only angles varying from zero (parallel planes) to 180° parallel planes facing opposite directions.

4.3.6.2 Algorithm description

We will start by considering the cases in which the two planes whose angles we want to measure are visible in a unique range image. Let us consider that the first plane can be described (using Equation (4.3.45)) by the constants a, b and c and the second plane can be described by the constants a', b' and c'. In this case if α is the angle between the two normals of the two planes, then:

$$cos(\alpha) = \frac{1 + aa' + bb'}{\sqrt{1 + a^2 + b^2}\sqrt{1 + a'^2 + b'^2}} \tag{4.3.53}$$

$$|sin(\alpha)| = \frac{\sqrt{(a - a')^2 + (b - b')^2 + (ab' - a'b)^2}}{\sqrt{1 + a^2 + b^2}\sqrt{1 + a'^2 + b'^2}} \tag{4.3.54}$$

If ϵ_{cos} and ϵ_{sin} are the errors in our estimate of $cos(\alpha)$ and $sin(\alpha)$ then the errors $\epsilon_{\alpha_{cos}}$ and $\epsilon_{\alpha_{sin}}$ in the estimate of α calculated using Equations (4.3.53) and (4.3.54)

are respectively given by Equations (4.3.55) and (4.3.56).

$$|\epsilon_{\alpha_{cos}}| = |\frac{-\epsilon_{cos}}{sin(\alpha)}| \tag{4.3.55}$$

$$|\epsilon_{\alpha_{sin}}| = |\frac{-\epsilon_{sin}}{cos(\alpha)}| \tag{4.3.56}$$

4.3.6.3 Error analysis

Although it is possible to establish coarse estimates of the maximum error in the estimate of α under some conditions, the determination of a general and accurate statistical model of the errors is very complicated due to the variation of the errors with the values of a, b, a' and b', as well as the many non-linear operations involved in the calculation of the estimate of α.

As before, we will use Monte Carlo simulations to build an accurate approximation of the error in the estimate of α for a particular set of conditions.

In our simulations the average error of the estimate of α was always near zero for N_{sq} at least equal to 15 which seems to indicate that the estimate is unbiased.

Figure 4.34 shows the maximum error, standard deviation, the average error, minus one standard deviation and the minimum error in the estimate of $cos(\alpha)$ as a function of N_{sq}. The graphs were obtained as the result of 100 simulations to each value of N_p for $a = 0.71$, $b = -2.5$, $c = 777.17$, $a' = -0.17$, $b' = -0.39$, $c' = 729.6$, $\alpha = 0.911$ rad and $\sigma_S = 0.1$ mm. The figure shows the typical order of magnitude of the errors we can expect in the estimate of $cos(\alpha)$.

Figure 4.35 shows the probability of the absolute value of the error of the estimate of $cos(\alpha)$ being bigger than δ_{LIM} (where δ_{LIM} corresponds to the maximum error of the estimate of $cos(\alpha)$ associated to a given probability). The graph was obtained as the result of 300 simulations in the same conditions of Figure 4.34 for $N_{sq} = 20$. The figure also shows an estimate of the probability calculated by approximating the result of the simulation by a normally distributed random variable of mean zero calculated to approximate the result of the simulation for values of probability around 0.04.

As we can see by the figure, the normal approximation is very close to the results of

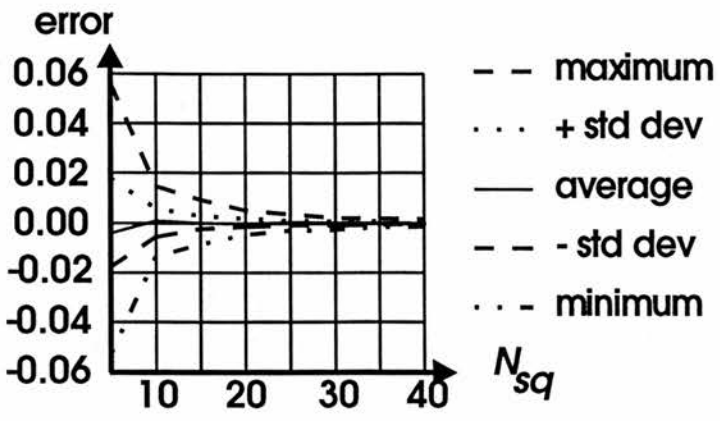


Figure 4.34: Maximum error, standard deviation of error, average error, minus standard deviation of error and minimum error in the estimate of $cos(\alpha)$. Result of one hundred simulations for each value of N_{sq}, $a = 0.71$, $b = -2.5$, $c = 777.17$, $a' = -0.17$ and $b' = -0.39$, $c' = 792.6$ and $\alpha = 0.911$ *rad.* Laser striper assumed to have pixels with size *0.5 mm* $\times$ *0.5 mm* and σ_S = *0.1 mm.*

the simulation.

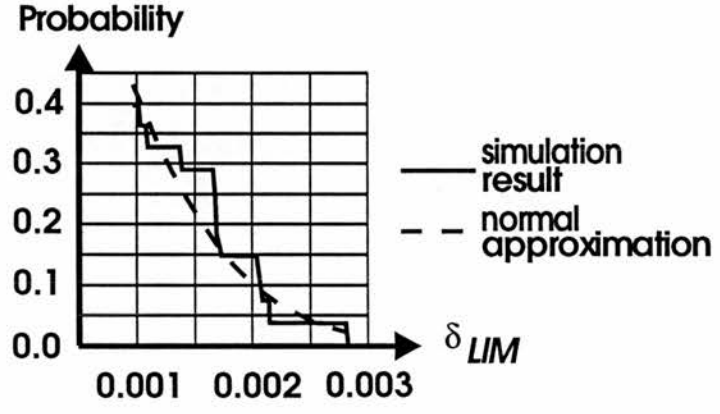


Figure 4.35: Probability of error in the estimate of $cos(\alpha)$ being bigger than δ_{LIM} and its approximation using a normally distributed random variable. Result of three hundred simulations for $N_{sq} = 20$, $a = 0.71$, $b = -2.5$, $c = 777.17$, $a' = -0.17$ and $b' = -0.39$ and $c' = 792.6$. Laser striper assumed to have pixels with size *0.5 mm* $\times$ *0.5 mm* and σ_S = *0.1 mm.*

Using the results shown in Figure 4.35 it is possible to determine the size of the uncertainty region (in this case a segment) associated with the measurement of the angle α for a given probability P of the actual value of α being outside the uncertainty region using Equation (4.3.55).

In the particular case of Figure 4.35, in which $\alpha = 0.911$ *rad*, according to Equation (4.3.55) the absolute error in the estimate of α corresponding to a given value of δ_{LIM} is equal to $\frac{\delta_{LIM}}{sin(0.911)}$. This means that the variation of δ_{LIM} shown in Figure 4.35

corresponds to a variation of the absolute error in the estimate of α from *1e-3 rad* to *4e-3 rad* approximately.

When we want to measure the angle between two planes that cannot be visible in a unique range image, we must consider the rotation of the vector corresponding to the normal of one of the two planes. These cases will occur when the planes face opposite directions with relation to the z axis of the laser striper.

Therefore, we can measure the angle between two planes in these cases using two range images taken in such a way that one of the range images corresponds to a rotation of the object being inspected around the x or y axis of the laser striper sensor. If we consider, for instance, a rotation of 180°around the x axis we will have:

$$cos(\alpha) = \frac{-1 + aa' - bb'}{\sqrt{1 + a^2 + b^2}\sqrt{1 + a'^2 + b'^2}} \tag{4.3.57}$$

$$|sin(\alpha)| = \sqrt{1 - (\frac{-1 + aa' - bb'}{\sqrt{1 + a^2 + b^2}\sqrt{1 + a'^2 + b'^2}})^2} \tag{4.3.58}$$

The errors associated with Equations (4.3.57) and (4.3.58) can still be modelled using Equations (4.3.55) and (4.3.56). It is important to observe that these equations are only valid if the precision of the manipulation of the object being imaged is such that the errors in the manipulation of the object are much smaller than the errors in the measurement of the angle between the two planes.

4.3.6.4 General properties of algorithm

As a final comment on the errors in the estimate of α we should say that, everything else being constant, the error will vary linearly with the value of σ_S. This fact it is illustrated by Figure 4.36 that shows the graph of the standard deviation of the error in the estimate of α versus the standard deviation of the error in the range measurement σ_S.

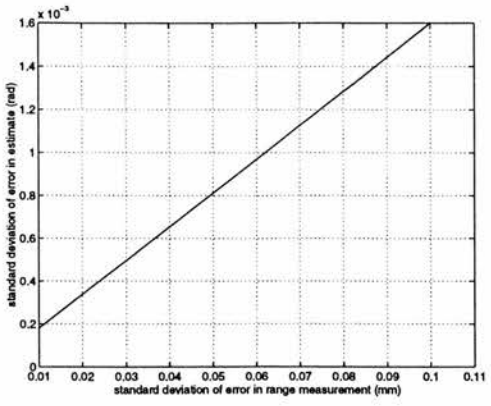


Figure 4.36: Standard deviation of error in the estimate of α versus the standard deviation of error in the range measurement σ_S. Result of 300 simulations for each value of σ_S under the same conditions of Figure 4.35 and with $N_{sq} = 20$.

4.3.7 Extraction of the angle between two lines corresponding to planes intersections

4.3.7.1 Objectives and motivation

The objective of this section to use of the least square fitting of planes to range image, discussed in Section 4.3.5, to measure the angle between lines corresponding to the intersection between two planes. We will only consider angles varying from 0° to 180° and we will also assume that the intersecting planes are always visible in the range image.

4.3.7.2 Algorithm description

Let us consider two planes respectively defined by the constants (a, b, c) and (a', b', c') (according to Equation (4.3.45)) such that the two planes are not parallel. The line corresponding to the intersection of these planes will be parallel to the vector $\mathbf{dir_{line}}$ given by:

$$\mathbf{dir_{line}} = \begin{bmatrix} b - b' \\ a' - a \\ a'b - ab' \end{bmatrix} \tag{4.3.59}$$

If ϵ_a, ϵ_b, $\epsilon_{a'}$ and $\epsilon_{b'}$ are the errors in the estimates of a, b, a' and b' respectively, the error in the estimate $\hat{\mathbf{dir}}_{\mathbf{line}}$, which we will call $\epsilon_{\mathbf{dir}}$ is given by:

$$\epsilon_{\mathbf{dir}} = \begin{bmatrix} \epsilon_b - \epsilon_{b'} \\ \epsilon_{a'} - \epsilon_a \\ \epsilon_{a'} b - \epsilon_a b' + \epsilon_b a' - \epsilon_{b'} a + \epsilon_{a'} \epsilon_b - \epsilon_a \epsilon_{b'} \end{bmatrix} \tag{4.3.60}$$

Let us now consider the vector $\mathbf{dir_{line}}^2$ corresponding to the intersection of two other planes distinctive from the planes defining $\mathbf{dir_{line}}$. In this case, the angle α between the two lines respectively parallel to $\mathbf{dir_{line}}$ and $\mathbf{dir_{line}}^2$ is such that $cos(\alpha)$ and $sin(\alpha)$ are:

$$cos(\alpha) = \frac{\mathbf{dir_{line}} \cdot \mathbf{dir_{line}}^2}{\|\mathbf{dir_{line}}\| \ \|\mathbf{dir_{line}}^2\|} \tag{4.3.61}$$

$$|sin(\alpha)| = \frac{\|\mathbf{dir_{line}} \times \mathbf{dir_{line}}^2\|}{\|\mathbf{dir_{line}}\| \ \|\mathbf{dir_{line}}^2\|} \tag{4.3.62}$$

4.3.7.3 Error analysis and general properties of the algorithm

Despite the fact that we can use a unique statistical model for the errors in the estimates of a, b, b' and a' (as discussed in Section 4.3.5), the construction of a general and accurate model of the errors in the estimate of α is very complicated due to the non-linear operations involved. Therefore, we will adopt the same procedure we did before and use Monte Carlo simulations to build accurate models of the errors in the estimate for a particular set of conditions.

Figures 4.37 illustrates the error in the estimate of $\frac{\mathbf{dir_{line}}}{\|\mathbf{dir_{line}}\|}$. The figure shows the variation of the error in the z component of $\frac{\mathbf{dir_{line}}}{\|\mathbf{dir_{line}}\|}$ as a function of N_{sq} (side in pixels of the square region of the image plane used in the least square fitting).

The figure was obtained as the result of 100 simulations for each value of N_{sq} for two intersecting planes defined (according Equation (4.3.45)) by the constants $a = 0.161$, $b = -0.123$,$c = 800.43$, $a' = -0.306$, $b' = 0.983$, $c' = 807.68$.

Although the figure shows only the error in the z component of $\frac{\mathbf{dir_{line}}}{\|\mathbf{dir_{line}}\|}$ the graphs of Figure 4.37 illustrate some general properties of the error of the estimate that we observed also in the previous section. As before the estimate is unbiased and the errors will decrease when N_{sq} increases.

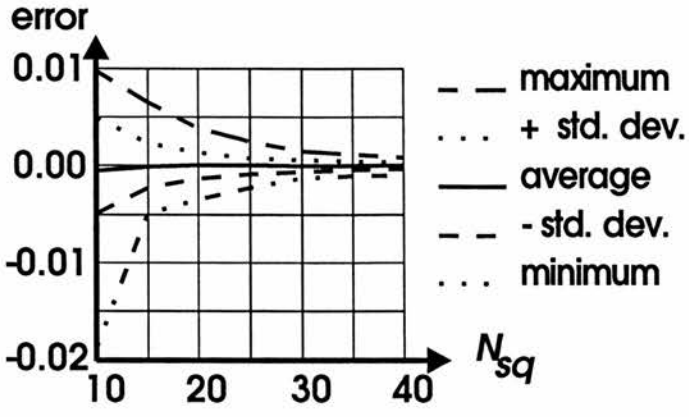


Figure 4.37: Error in the estimate of the z component of $\frac{\mathbf{dir_{line}}}{\|\mathbf{dir_{line}}\|}$. Result of one hundred simulations to each value of N_{sq}, $a = 0.161$, $b = 0.123$, $c = 800.43$, $a' = -0.306$ and $b' = 0.983$ and $c' = 807.68$. Laser striper assumed to have pixels with size *0.5 mm* × *0.5 mm* and σ_S = *0.1 mm*.

Figure 4.38 shows the error in the estimate of $sin(\alpha)$ when we consider two lines such that the first one is defined by the same two intersecting planes used in the simulations of Figure 4.37 and the second line is defined by the intersection of two planes defined (according to Equation (4.3.45)) by: $a = -0.036$, $b = -2.572$, $c = 785.06$, $a' = -0.360$, $b' = -0.965$ and $c' = -792.403$ (α = *0.630 rad*).

The figure illustrates the typical behaviour of the error in the estimate of $sin(\alpha)$: the estimate is always unbiased and its errors decrease when N_{sq} increases.

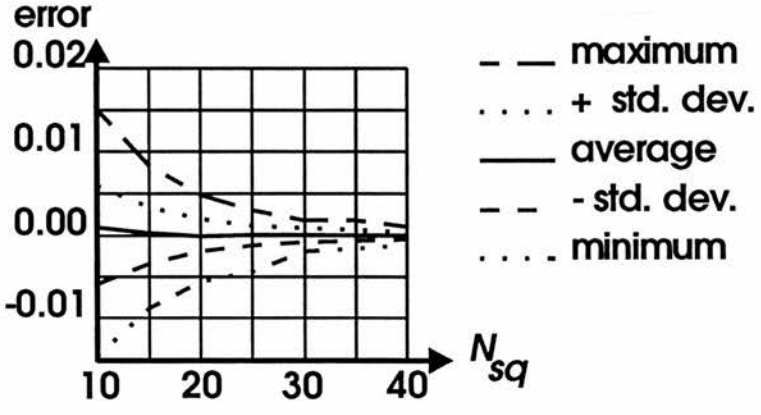


Figure 4.38: Error in the estimate of the sine of the angle between two lines corresponding to the intersection of planes. Result of one hundred simulations to each value of N_{sq} for intersecting planes defined by the constants $a = 0.161$, $b = -0.123$, $c = 800.43$, $a' = -0.307$, $b' = 0.983$, $c' = 807.68$, $a_2 = -0.036$, $b_2 = -2.572$, $c_2 = 785.1$, $a'_2 = -0.36$, $b'_2 = -0.965$, $c'_2 = -792.4$. Angle between lines equal to *0.63 rad*. Laser striper assumed to have pixels with size *0.5 mm* × *0.5 mm* and σ_S = *0.1 mm*.

To determine the uncertainty region associated with the estimate of α for $N_{sq} = 20$

we proceeded with 300 simulations in the same conditions of Figure 4.38 and observed the probability density function of having an absolute error in the estimate of $sin(\alpha)$ bigger than δ_{LIM}.

The result of the simulations is shown in Figure (4.39), which also shows the graph of a local approximation of the probability density function created to minimise the errors for probabilities around 0.01.

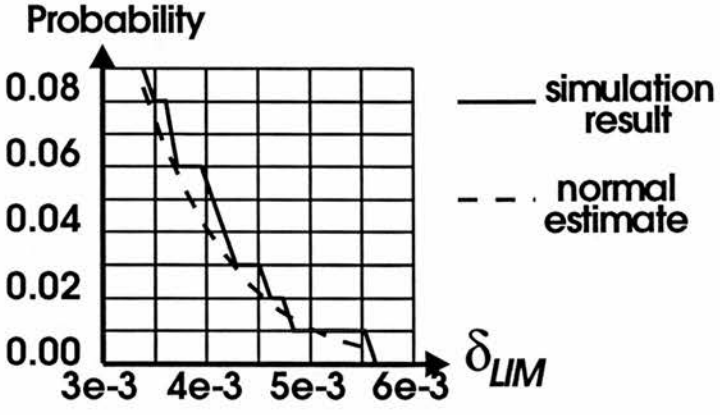


Figure 4.39: Probability of error in the estimate of $sin(\alpha)$ being bigger than δ_{LIM} and its approximation using a normally distributed random variable. Result of three hundred simulations for $N_{sq} = 20$ and the same simulation conditions of Figure 4.38. Laser striper assumed to have pixels with size $0.5\ mm \times 0.5\ mm$ and $\sigma_S = 0.1\ mm$.

The graph of Figure 4.39 can be used to determine the size of the uncertainty region in the estimate of α associated with a given probability of the absolute error in the estimate of α being bigger than a given value.

In the case of Figure 4.39, according to Equation (4.3.56), the absolute error in the estimate of α is equal to $\frac{\delta_{LIM}}{cos(0.63)}$. This means that the variation of δ_{LIM} from $3e-3$ to $6e-3$ in Figure 4.39 corresponds to absolute errors in the estimate of α varying from approximately *4e-3 rad* to *7e-3 rad.*

In the discussion so far, we implicitly assumed that all the planes necessary to determine the two lines whose angle we want to measure can be visible in a unique range image. When this is not possible we will have to consider the eventual rotations of the object. This fact does not create any problem in the determination of the uncertainty zones using the Monte Carlo simulation, because we will just have to consider the necessary rotations in the simulations.

Besides all that was already discussed about errors in the estimate of α, another important property we also verified in the simulation is that, everything else being constant, the standard deviation of the errors will vary linearly with the standard deviation of the sensor noise σ_S. This fact is illustrated by Figure 4.40 that shows the graph of the standard deviation of the error of the estimate of α versus the standard deviation of the error in the range measurement σ_S. The graph was obtained as a result of 300 simulations for each value of σ_S under the same conditions of Figure 4.38.

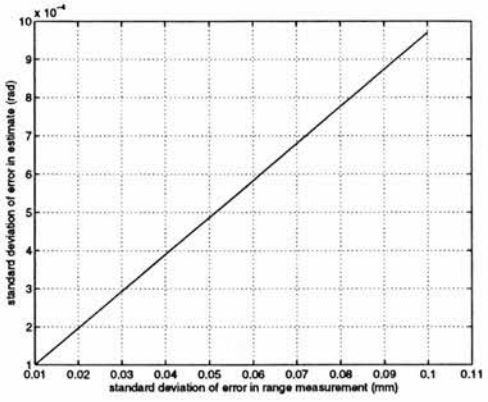


Figure 4.40: Standard deviation of error in the estimate of α versus the standard deviation of error in the range measurement σ_S. Result of 300 simulations for each value of σ_S under the same conditions of Figure 4.38 and with $N_{sq} = 35$.

4.4 General model of sensitivity and reliability using uncertainty regions

In order to build a general model of sensitivity and reliability based on uncertainty regions we will consider two different tolerance zones:

- **Conventional tolerance zone** (size T_z): a set of manufactured features is considered to be within tolerance if all the manufactured features are inside a tolerance zone of size T_z. The shape of the conventional tolerance zone is determined by offsetting the geometrical elements corresponding to the measured features.

- **Designed tolerance zone** (size T_z^m): corresponds to the offset region in which all the manufactured features are expected to be inside given the accuracy of the

manufacturing process chosen during the design phase. The designed tolerance zones are always smaller than the conventional tolerance zones.

The size of the two tolerance zones are related by Equation (4.4.63) where K_c is a constant bigger than or equal to one that is called *process capability index.*

Figure 4.41 shows the two tolerance zones in the case in which the geometric element being offset is a straight line.

$$T_z^m = \frac{T_z}{K_c} \tag{4.4.63}$$

The idea behind the definition of two tolerance zones is that to achieve the necessary accuracy (T_z) in the final manufactured object (that is determined basically by the functional requirements of the object we are inspecting) we use a manufacturing process able to achieve a better accuracy (T_z^m).

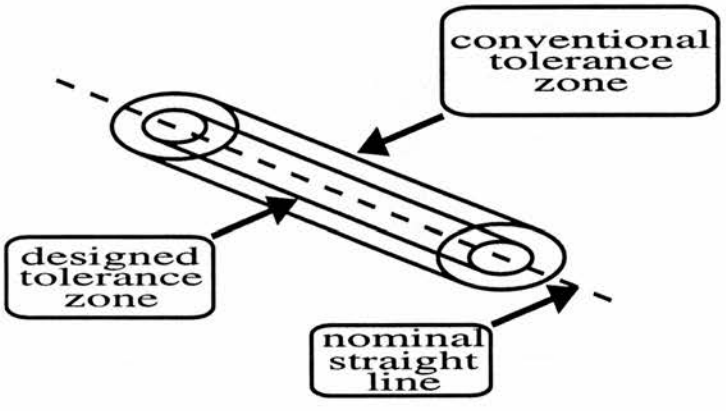


Figure 4.41: Conventional tolerance zone and designed tolerance zone when the geometric element being offset is a straight line.

It is possible to understand this procedure if we consider the errors in the manufacturing processes as random variables with mean zero such that the bigger the amplitude of the error the smaller the probability of the error to occur. This means that, for a given size of tolerance zone T_z and a given manufacturing process (corresponding to a value of T_z^m), we can associate a corresponding probability of the manufacturing process to produce an object out of tolerance.

For a fixed size of tolerance zone T_z the probability of producing an object out of tolerance decreases when the process capability index (K_c) increases and, consequently, the accuracy of the manufacturing process increases.

The objective of this section is to determine the relation between the sizes of the two tolerance zones (T_z and T_z^m), the statistical model of the errors of the feature extraction technique used and the reliability and sensitivity of the inspection procedure considered.

To achieve this objective we will start by defining a numerical way of measuring the reliability and sensitivity of an inspection procedure. We will measure the reliability and sensitivity of an inspection procedure using two different probabilities:

- the probability P_f (false alarm) of rejecting a manufactured element that was manufactured within the designed tolerance (tolerance zone of size T_z^m) and
- the probability P_n of not detecting that the element being inspected was manufactured out of the conventional tolerance (tolerance zone of size T_z).

Given the two tolerance zones and the desired reliability (expressed by the probability P_f) and the desired sensitivity (expressed by the probability P_n) we want an inspection procedure diagnosis I_p such that:

- There is a probability P_f of classifying a set of manufactured features $F_a = \{f_i^a\}_{(i=1,N_{feat})}$ as out of tolerance if all the features f_i^a belonging to F_a are inside the designed tolerance zone (size T_z^m).
- There is a probability P_n of not detecting that some of the manufactured features f_i^a are outside the conventional tolerance zone (size T_z).

The main idea here is that we want to be sure that objects manufactured with shape errors smaller than the designed ones will be almost surely accepted and objects manufactured with shape errors bigger than that allowed by the functional requirements will be almost surely rejected. Objects with shape errors between these two extremes might be rejected or accepted, and the probability of rejection increases as the size of the errors increases.

Now let us consider a feature extraction procedure such that, given the measured feature f_i^m, we have a probability P_i that the corresponding manufactured feature (f_i^a) will be outside an uncertainty region R_i with center of mass f_i^m (see Figure 4.42).

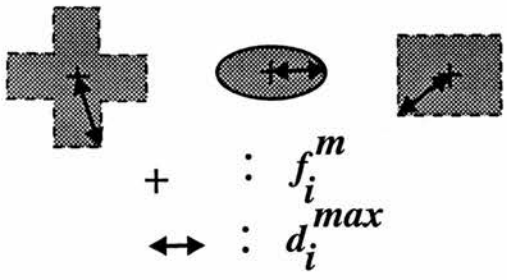


Figure 4.42: Value of d_i^{max} for three different shapes of 2D uncertainty region: cross, rectangle and ellipse.

In this case, if we define d_i^{max} as the maximum distance between a point x in the region R_i and the center of mass of the region R_i (f_i^m), we will have a probability at most equal to P_i of the actual feature (f_i^a) being at a distance d bigger than d_i^{max} of the measured feature.

$$d_i^{max} = max_{(R_i)} \ \|x - f_i^m\| \tag{4.4.64}$$

If the actual feature (f_i^a) is inside the designed tolerance zone (T_z^m), the distance between the actual feature f_i^a and the center of the designed tolerance zone is at most equal to ($\frac{T_z^m}{2}$), this means that we will have a probability smaller than P_i that the measured feature f_i^m will be outside a tolerance zone of size bigger than T_z^{min} given by (see Figure 4.43):

$$\frac{T_z^{min}}{2} = d_i^{max} + \frac{T_z^m}{2} \tag{4.4.65}$$

Analogously, we will have a probability smaller than P_i of the measured feature f_i^m being inside a tolerance zone of size smaller than T_z^{max} (see Equation (4.4.66)) when the actual feature (f_i^a) is outside the tolerance zone (T_z)

$$\frac{T_z^{max}}{2} = -d_i^{max} + \frac{T_z}{2} \tag{4.4.66}$$

Therefore, if we define a *test tolerance zone* of size T_{test} such that T_{test} is smaller than T_z^{max} and T_{test} is also bigger than T_z^{min}, we will have a probability smaller than P_i of a measured feature f_i^m to be outside the test tolerance zone when the actual feature f_i^a is inside the designed tolerance zone. Furthermore, we will have a probability smaller

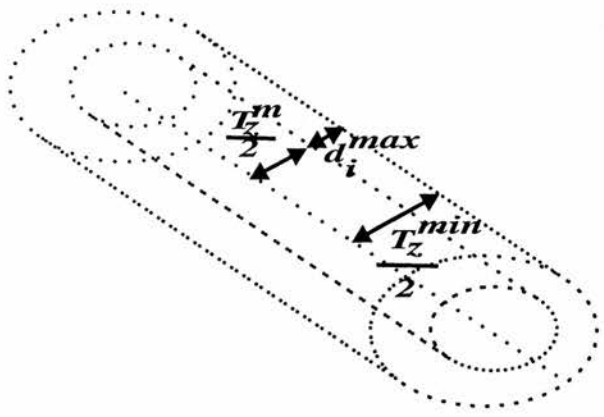


Figure 4.43: Illustration of designed tolerance zone (size T_z^m) and minimum test tolerance zone (size T_z^{min}) when geometric element being offset is a straight line.

than P_i of a measured feature f_i^m being inside the test tolerance zone when the actual feature f_i^a is outside the conventional tolerance zone.

For a given value of P_i the smallest tolerance zone we can measure corresponds to the case in which T_z^{min} is equal to T_z^{max} and to T_{test}, this implies that:

$$T_z = 4d_i^{max}\frac{K_c}{K_c - 1} \tag{4.4.67}$$

Equation (4.4.67) represents a fundamental relation in the modelling of the sensitivity and reliability of inspection procedures. The equation makes it possible to determine the value of d_i^{max} we must have in order to inspect a given feature with a tolerance zone of size T_z and process capability index K_c.

Because d_i^{max} can be related to the probabilities P_f and P_n, the sensor errors and feature extraction parameters, Equation (4.4.67) allows the determination of the sensor and feature extraction parameters necessary to accomplish the desired inspection with the desired reliability and sensitivity. Examples of the use of this equation in particular implementations of inspection procedures are given in Section 4.6.

From now on, we will call the $T_z^{test}(T_{test}, Pose, Scale)$ a *test tolerance zone* of size T_{test} such that the geometrical element offset to create the tolerance zone has pose $Pose$ and scale $Scale$. Where the size T_{test} of the test tolerance zone is determined according to the probabilities:

- P_f^{feat} is the probability of a measured feature f_i^m being outside the test tolerance zone (size T_{test}) when the actual feature f_i^a is inside of the designed tolerance zone (size T_z^m) and
- P_n^{feat} is the probability of a measured feature f_i^m to be inside the test tolerance zone (size T_{test}) when the actual feature f_i^a is outside of the conventional tolerance zone (size T_z).

We will also call the geometrical element offset that creates the tolerance zone the *tolerance zone center.*

Equation (4.4.67) is only valid when P_f^{feat} is equal to P_n^{feat}. The equation can easily be extended for the cases in which P_f^{feat} and P_n^{feat} are not the same. Let us consider two different sizes of uncertainty regions d_n^{max} and d_f^{max} such that:

- d_n^{max} is the maximum distance of the uncertainty regions for a probability P_n^{feat} of the actual value being outside the uncertainty region and
- d_f^{max} is the maximum distance of the uncertainty region for a probability P_f^{feat} of the actual value being outside the uncertainty region.

In this case we will have:

$$\frac{T_{test}}{2} = \frac{T_z}{2} - d_n^{max} \tag{4.4.68}$$

$$T_z = 2\frac{(d_n^{max} + d_f^{max})K_c}{K_c - 1} \tag{4.4.69}$$

Equation (4.4.69) demonstrates clearly a fundamental property of the inspection process that we will see again in Chapter 6. For a fixed set of conditions (size of tolerance zone, process capability index, viewing conditions, shape of feature and feature extraction technique) we can only increase the reliability (value of d_f^{max}) if we decrease the sensitivity (value of d_n^{max}) and vice-versa.

Another interesting property of the Equations (4.4.67) and (4.4.69) is the fact that T_z approaches infinity when K_c approaches one. This happens because when K_c approaches one T_z^m approaches T_z. Thus, it is not possible to satisfy Equation (4.4.65) and (4.4.66) simultaneously for finite values of d_n^{max} and d_f^{max}.

The inspection in the case in which K_c is equal to one would only be possible if we did not have errors in the feature extraction or, in other words, if the uncertainty regions collapsed to a point.

It is important to observe that Equations (4.4.67) and (4.4.69) overestimate the size of the tolerance zone we are able to measure for given values of P_f^{feat} and P_n^{feat}. This happens because both equations do not take into account the shape of the uncertainty regions and their position with relation to the test tolerance zone.

So far, we have only been concerned with establishing the relation between the uncertainty regions of the feature extraction process and the probabilities P_f^{feat} and P_n^{feat}. Our final objective is to relate the uncertainty regions with the probabilities P_f and P_n of errors during the inspection of a geometric element corresponding to a set of measured features F_m

In order to do so, let us suppose that for a given test tolerance zone $T_z^{test}(T_{test}, Pose, Scale)$ the inspection will consider the manufactured geometric element within tolerance only if the number of measured features f_i^m outside the test tolerance zone is smaller than or equal to a chosen value F_{out}. We will also assume that we have T_{out} features outside the conventional tolerance zone.

Furthermore, let us also suppose that the errors in the extraction of the features are independent, *i.e.* the errors in the extraction of a given feature f_i^m are not correlated to the errors in the extraction of any other feature in the set of measured features F_m. This is a reasonable assumption as long as the features are extracted using local operators in such a way that a given region of data is not used to extract more than one feature of the set of measured features F_m. For instance, this is true of the case in which we extract edge points along a line using the algorithm of Section 4.3.1 in such a way that different regions of the data set are used to extract each of the edge points.

Under these assumptions, the probability P_f is related to the number of measured features N_{feat}, the probability P_f^{feat} and F_{out} by Equations (4.4.70), (4.4.71) and (4.4.72).

$$B_{bin}(p,k,n) = \frac{n!}{k!(n-k)!}p^k(1-p)^{(n-k)} \tag{4.4.70}$$

$$P_{bin}(p,x,n) = \sum_{k=0}^{k=x} B_{bin}(p,k,n) \tag{4.4.71}$$

where: p is a real number between 0 and 1, n is a positive integer number and x is an integer number smaller than or equal to n

$$P_f = 1 - P_{bin}(P_f^{feat}, F_{out}, N_{feat}) \tag{4.4.72}$$

In a similar way, if we assume that T_{out} bigger than F_{out}, the probability P_n is given by Equation (4.4.73).

$$P_n = \sum_{k=0}^{k=F_{out}} B_{bin}(1-P_n^{feat}, k, T_{out}) P_{bin}(P_f^{feat}, F_{out}-k, N_{feat}-T_{out}) \tag{4.4.73}$$

where F_{out} is supposed to be smaller than T_{out}

Equations (4.4.72) and (4.4.73) show again the trade-off between the sensitivity and the reliability of the inspection procedure. For fixed values of P_f^{feat} and P_n^{feat} it is only possible to reduce P_f by increasing F_{out}, *i.e.* increasing the probability P_n. This fact is illustrated by Table 4.7 that shows the variation of P_f and P_n with F_{out} when we have $N_{feat} = 60$, $T_{out} = 4$ and $P_f^{feat} = P_n^{feat} = 0.01$.

As we will illustrate in Section 4.6, the value of T_{out} can be associated with the minimum size of isolated deformation in the geometric element we want be able to detect with probability not smaller than $1 - P_n$.

The final diagnosis of an inspection procedure, as we defined, is produced by comparing the position of the measured features with relation to the test tolerance zone. This means that, besides the feature extraction, another essential step in the inspection is the determination of the shape, size and position of the tolerance zone associated with the geometric element we want to inspect. This subject is discussed in the next section in which we define formally the diagnosis of an inspection procedure.

F_{out}	P_f	P_n
0	4.53 e-1	5.70 e-9
1	1.21 e-1	2.26 e-6
2	2.24 e-2	3.39 e-4
3	3.12 e-3	2.26 e-2
4	3.46 e-4	5.82 e-1

Table 4.7: Variation of P_f and P_n with the value of F_{out}. Result corresponding to $P_f^{feat} = P_n^{feat} = 0.01$, $N_{feat} = 60$ and $T_{out} = 4$.

4.5 Inspection procedure diagnosis

In this section we are concerned with the formal definition of tolerance zones and the inspection diagnosis in a inspection procedure. Although the ideas we are going to expose agree with the common practice described in the standards in many cases, our main objective in this section is the adaptation of Requicha's theory of tolerancing for the inspection of mechanical parts.

In order to make possible the formal definition of the final inspection diagnosis, we will start by defining the inspection diagnosis of a set of measured features F_m, that we will call $I_p(F_m, T_{test}, Pose, Scale, F_{out})$, when we consider a test tolerance zone $T_z^{test}(T_{test}, Pose, Scale)$ as :

- $I_p(F_m, T_{test}, Pose, Scale, F_{out})$ is equal to one when the number of features in F_m outside the test tolerance zone $T_z^{test}(T_{test}, Pose, Scale)$ is smaller than or equal to F_{out} and
- $I_p(F_m, T_{test}, Pose, Scale, F_{out})$ is equal to zero when the number of features in F_m outside the test tolerance zone $T_z^{test}(T_{test}, Pose, Scale)$ is bigger than F_{out}

In the case of shape inspection we want to check the shape of a geometrical element, (*e.g.* the straightness of a manufactured segment, the planarity of a manufactured planar patch or the circularity of hole). This means that we are concerned neither with the size of element we want to inspect (*e.g.* the segment length, the planar patch area or the circle radius) nor with its pose(*e.g.* segment's endpoints positions, planar patch orientation, the circle center position).

Following Requicha's ideas and the standards in some cases we will define the test tolerance zones by:

1. offsetting the infinite geometric element corresponding to the particular geometric element we want to inspect (*e.g.* in the case of a segment of line we will offset an infinite straight line, in the case of a planar patch we will offset an infinite plane, a circle in the case of a circle)

2. allowing variations in the scale of the geometric element we are offsetting (*e.g.* in the case of a circle we will consider test tolerance zones created by offsetting circles of different radius)

3. allowing variations in the pose of the test tolerance zone (*e.g.* in the case of a line we will vary the line position and slope, in the case of a circle we will vary the position of the circle's center)

If $I_p(F_m, T_{test}, Pose, Scale, F_{out})$ specifies the inspection diagnosis of the set of measured features F_m considering a test tolerance zone created according to the rules announced above, we will define the shape inspection diagnosis I_p^{shape} as:

$$I_p^{shape} = \begin{cases} 1 - P_n, & \exists\, Pose\ \exists\, Scale \mid I_p(F_m, T_{test}, Pose, Scale, F_{out}) = 1 \\ P_f, & \text{otherwise} \end{cases} \tag{4.5.74}$$

where: P_f and P_n are the same as the previous section and are determined using Equations (4.4.72) and (4.4.73).

I_p^{shape} measures our certainty about the fact that the set of actual features corresponding to the measured features F_m is within tolerance. That is, I_p^{shape} near to one means that we are almost completely sure that the manufactured element being inspected is within tolerance and I_p^{shape} near to zero means that we are almost sure that the manufactured element is out of the designed tolerance.

In the case of size inspection, we are still not concerned with the pose of the actual feature, but now we are interested in the actual size of the feature. This means, that we will have two big differences in the determination of the test tolerance zones:

1. the tolerances zones will be created by offsetting the geometrical element we want to inspect (*e.g.* in the case of a segment we will offset the segment itself and in the case of a planar patch we will offset the planar patch itself)

2. we will allow only variations in the pose of the geometrical element we offset to create the test tolerance zone, *i.e.* we will no longer allow variations in scale.

Analogously to the case of I_p^{shape}, we will define the size inspection diagnosis (I_p^{size}) as:

$$I_p^{size} = \begin{cases} 1 - P_n, & \exists\ Pose \mid I_p(F_m, T_{test}, Pose, S_N, F_{out}) = 1 \\ P_f, & \text{otherwise} \end{cases} \tag{4.5.75}$$

where : the value of I_p^{size} has the a similar meaning to the value of I_p^{shape} and S_N is the nominal size of the element being inspected.

From all that was discussed so far, the shape inspection procedure diagnosis I_p^{shape} and the size inspection procedure diagnosis I_p^{size} involve a non-linear minimisation process in which we try to find a pose and a scale (only a pose in the case of size inspection procedure) such that the number of measured features outside the test tolerance zone $T_z^{test}(T_{test}, Pose, Scale)$ is minimum (or at least not bigger than F_{out}).

For doing so, in the case of shape inspection procedure, we will take the following steps:

1. Find a first approximation of the tolerance zone center by applying a least square procedure to the set of measured features F_m. This process yields a pose $Pose_0$ and scale $Scale_0$ associated with the least square approximation of the geometric element.

2. Verify the number of features of F^m that are outside the tolerance zone defined using the least square approximation already obtained. If the number is bigger than F_{out}, consider other tolerance zones with different pose $Pose$ and scale $Scale$ such that $Pose$ and $Scale$ are in a suitable neighbourhood of $Pose_0$ and $Scale_0$

In the case of size inspection procedures we will follow the same procedure with the difference that in this case the $Scale$ of the element being inspected is already determined and, consequently, there will not be variations in the $Scale$ of the element being offset to determine the test tolerance zones.

Finally, in the case of pose inspection, the scale *Scale* and the pose *Pose* of the geometric element we want to inspect are already defined. Therefore, one could expect that we would have to consider just one test tolerance zone.

However, this is not true due to the fact that the pose of the test tolerance zone is determined in relation to a specific datum that corresponds to a set of features in the object being inspected. This means that we have to take into account the uncertainty in the position of the datum as well.

It is important to observe that when we consider the uncertainties in the datum position we are actually departing from the usual behaviour adopted by many manufacturers. In most of the cases the datums are physically determined by the contact between some surfaces (edges or vertices) of the object and the surfaces (edges or vertices) of a mechanical template.

This means that the surfaces (edges or vertices) corresponding to the datums are implicitly accepted as in the right position, *i.e.* the inspection procedure does not make any considerations about possible errors in the datum position.

In many cases the size of the uncertainty regions associated with the errors in the datum(s) pose are negligible due to the great number of features used in the datum determination. In these cases, we can ignore the errors caused by the uncertainty in the datum position.

In order to consider the errors caused by the datum extraction, let us start by considering that the uncertainty in the datum extraction is such that the error in the pose $Pose_0$ of the tolerance zone corresponds to a random variable $\delta Pose$ with a mean of zero such that we have a probability P_{out} of the right value of pose being outside an uncertainty region R_P. In this case, we will define the pose inspection procedure I_p^{pose} of the geometrical element corresponding to the set of measured features F_m as:

$$I_p^{pose} = \begin{cases} (1-P_n)(1-P_{out}), & \forall\, Pose \in R_P,\ I_p(F_m, T_{test}, Pose, S_N, F_{out}) = 1 \\ P_f(1-P_{out}), & \forall\, Pose \in R_P,\ I_p(F_m, T_{test}, Pose, S_N, F_{out}) = 0 \\ P_{OK}, & \text{otherwise} \end{cases} \tag{4.5.76}$$

where: S_N is the nominal scale of the geometric element we want to inspect

If $f_{\delta Pose}(.)$ is the probability density function of the random variable $\delta Pose$ and S_f is the subset of the uncertainty region R_P (volume, area or length depending on the dimension of the uncertainty region R_P) for which $I_p(F_m, T_{test}, Pose, S_N, F_{out})$ is equal to zero, the value of P_{OK} can be approximated by Equation (4.5.77):

$$P_{OK} = (P_f + P_n - 1) \int_{S_f} f_{\delta Pose}(Pose) dPose + (1 - P_n)(1 - P_{out}) \qquad (4.5.77)$$

The exact calculation of P_{OK} using Equation (4.5.77) is not possible as we cannot know exactly either the function $f_{\delta Pose}$ or the subset S_f. However, it is possible to calculate an approximation of the value of P_{OK} in many situations using Equation (4.5.78). This equation can be obtained by dividing R_P in many small subsets and associating each subset i of R_P to a probability $\Delta Prob^i$ given by Equation (4.5.79).

$$P_{OK} = (P_f + P_n - 1) \sum_{S_f} \Delta Prob^i + (1 - P_n)(1 - P_{out}) \qquad (4.5.78)$$

$$\Delta Prob^i = {f_{\delta Pose}}^i \ \Delta Pose^i \qquad (4.5.79)$$

where :

- ${f_{\delta Pose}}^i$ is the value of $f_{\delta Pose}$ at the center of the subset i of R_P and
- $\Delta Pose^i$ is size of the subset i of R_P (length, area or volume depending on the dimension of R_P)

In the cases in which the size of the uncertainty region R_P is much smaller than T_{test} (at least ten times smaller) it is possible to ignore the uncertainties in the determination of the pose and consider just a unique tolerance zone of pose $Pose_0$.

It is important to observe that the definition of the I_p^{pose} given by Equation (4.5.76) is very demanding and not ideal. Geometric elements manufactured within the desired tolerance might be refused when the uncertainty in the determination of the datum position is too big with relation to the size of the tolerance zone. This means that Equation (4.5.76) should be used very carefully and we should always try to reduce the uncertainty in the determination of the datum pose as much as possible.

When we are inspecting a global feature, such as the angle between the two normals or the radius of a circle, we do not have to worry about the shape, scale and position of the test tolerance zone.

The inspection of global features can be seen as an special case in which the tolerance zone is a segment and the center of the test tolerance zone is point. This means that the inspection diagnosis (that we will call I_p^{global} and define in a similar way to I_p^{shape} and I_p^{size}) can be obtained just by comparing two numbers.

4.6 Implementation of common inspection procedures

This section intends to exemplify the ideas described in the previous sections for inspection procedures. Our main objectives are twofold:

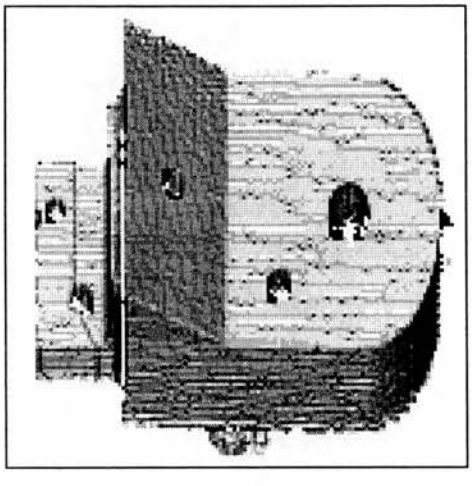


Figure 4.44: Range image of the *widget*: 128×128 pixels of size $2\ mm \times 1\ mm$ obtained using a sensor with $\sigma_S = 0.1\ mm$.

- the implementation of inspection procedures for checking some of the most common tolerancing described in the ANSI standards and
- to illustrate the practical use of Requicha's tolerancing theory.

Taking into account these objectives, we will describe the implementation of inspection procedures for checking the following tolerances:

- FORM: straightness, flatness and circularity.

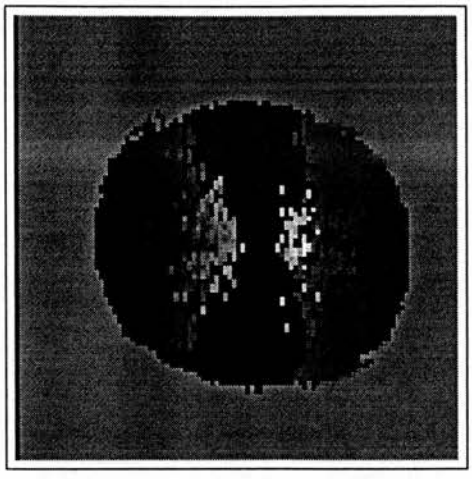


Figure 4.45: Range image of the hole H (see Figure 4.46) in *widget*: 100 × 100 pixels of size *0.28 mm* × *0.30 mm* obtained using a sensor with $\sigma_S = 0.1$ *mm*.

- ORIENTATION: angularity, perpendicularity and parallelism
- SIZE: segments of line and radius of circles
- LOCATION: segments of line and circles

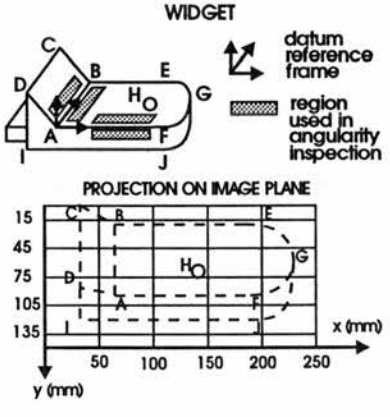


Figure 4.46: Draft of widget and projection on image plane of Figure 4.44

The details concerning the implementation of the different inspection procedures, the performance of the inspection procedures in simulated tests, as well as their application to the inspection of an actual object (the *widget* - see Figures 4.44, 4.45 and 4.46) are shown in the next sections.

4.6.1 Straightness Inspection

Straightness inspection is the verification of the shape of a curve. It aims to check if the deviations between the shape of the curve and the shape of a straight line are within the desired tolerances. We will use the term straightness inspection to designate the inspection of curves that are created by the intersection of two manufactured planar surfaces.

Taking into account the specification of the evaluation of straightness in the ANSI and BS standards, that are based in the concept of minimum zone appearing in the ISO, we will implement the straightness inspection procedure using the I_p^{shape} diagnosis described in the previous section.

We will also assume that the set of measured features F_m is a set of 3D edge points extracted using the algorithms described in Section 4.3.1 and Section 4.32. Because of the smaller accuracy in the extraction of step edge points, in this section we will be concerned only with the cases in which F_m is a set of roof edge points, *i.e.* we will assume that the object is positioned with relation to the sensor in such a way that the two intersecting manufactured surfaces are visible in the range image.

As the geometric element we are inspecting is a segment of line, the test tolerance zone $T_z^{test}(T_{test}, Pose, Size)$ is an infinite circular cylinder with radius T_{test} whose axis is an infinite straight line. We will define the pose $Pose$ of the axis of the test tolerance zone using a point with coordinates (x_p, y_p, z_p) belonging to the axis and two angles (θ and ϕ) determining the axis direction.

The implementation of the I_p^{shape} procedure involves the determination of the pose $Pose$ of the axis of the tolerance zone that minimises the number of measured edge points outside the tolerance zone.

Therefore, the determination of the pose $Pose$ of the axis of the test tolerance zone ($Pose = [x_p, y_p, z_p, \theta, \phi]$) can be done by minimising the number of measured edge points at a distance bigger than $\frac{T_{test}}{2}$ from the axis of the test tolerance zone. We will call this minimisation procedure as *approach 1.*

Another possibility in the determination of the pose of the axis of the test tolerance

zone, that we will call *approach 2*, is the minimisation of the maximum distance between the axis of the test tolerance zone and the edge points in the set of measured features. We shall comment the difference between the two approaches to positioning the axis of the test tolerance zone later.

We implemented the minimisation process for determining the $Pose$ of the test tolerance zone as search in a neighbourhood of the pose $Pose_m$ ($Pose_m = [x_p^m, y_p^m, z_p^m, \theta^m, \phi^m]$) of the straight line corresponding to the least square approximation of the set of measured edge points F_m (see Equations (4.3.33) to (4.3.39)).

The optimal value of pose is determined by exhaustive search in a discrete grid with $(N_{pose})^5$ elements in which each element $Pose_{(i,j,k,l,w)}$ was defined as:

$$Pose_{(i,j,k,l,w)} = [x_p^i, y_p^j, z_p^k, \theta^l, \phi^w] \tag{4.6.80}$$

$$x_p^i = x_p^m - \frac{N_{pose}}{2}\Delta_{loc} + i\Delta_{loc} \tag{4.6.81}$$

$$y_p^j = y_p^m - \frac{N_{pose}}{2}\Delta_{loc} + j\Delta_{loc} \tag{4.6.82}$$

$$z_p^k = z_p^m - \frac{N_{pose}}{2}\Delta_{loc} + k\Delta_{loc} \tag{4.6.83}$$

$$\theta^l = \theta^m - \frac{N_{pose}}{2}\Delta_{angle} + l\Delta_{angle} \tag{4.6.84}$$

$$\phi^w = \phi^m - \frac{N_{pose}}{2}\Delta_{angle} + w\Delta_{angle} \tag{4.6.85}$$

Where:

- i, j, k, l and w are integers that can vary from 0 to N_{pose}
- Δ_{loc} is an increment value defined (as a rule of thumb) in such a way that $N_{pose}\Delta_{loc}$ is equal to twice the maximum distance between the least square approximation of F_m and the features in F_m and
- Δ_{angle} is defined (as a rule of thumb) in such a way that $N_{pose}\Delta_{angle}$ is equal to *0.25 rad.*

As an example, let us consider the inspection of the line (segment AB in Figure 4.46) corresponding to the intersection of two manufactured planar surfaces in such viewing conditions that:

- the angles ϕ_1 and ϕ_2 between the normals of the intersecting planes ($ABCD$ and $ABEGF$ in Figure 4.46) and the viewing direction are respectively equal to *0.66 rad* and *0.51 rad*,
- the angle θ between the 2D edge line and the y axis of the sensor reference frame is *0.02 rad*,
- the angle ϵ between the 3D edge line and the image plane is *0.51 rad* and
- the projections of the two intersecting planes on the image plane correspond to a rectangular area of size *20 mm* by *60 mm* approximately

If we also assume that we are using a laser striper with $\sigma_S = 0.1\ mm$ and pixels of size $2\ mm \times 1\ mm$, the biggest possible value of N_{pixel} is 10.

Under these conditions, by applying Equations (4.3.6) to (4.3.11) to the result of 400 simulations, we conclude that: for probability $P = 0.01$ the uncertainty region associated with the 3D edge points can be modelled as planar region on the plane xz limited by an ellipse of main axis equal to *0.32 mm* and *1.06 mm* whose smallest main axis is approximately parallel to the z axis of the sensor reference frame.

Therefore, for $P_f^{feat} = P_n^{feat} = 0.01$, the size of the minimum tolerance zone we can inspect will vary with K_c (see Equation (4.4.67)) as it is shown in Figure 4.47. As we can see in the figure the value of T_z decreases when K_c approaches infinity. The smallest possible value of T_z (when K_c equals to infinity) is $T_z = 4.24\ mm$ for which we have $T_{test} = 2.12\ mm$.

In order to observe the result of the application of the straightness inspection procedure, we proceeded with 200 simulations of the application of the inspection procedure to synthetically created range images of two intersecting planes. The synthetic images were created using the same conditions of Figure 4.47, assuming the pixel size equal to $2\ mm \times 1\ mm$. Furthermore, we used $T_z = 4.24\ mm$ (corresponds to K_c equal to infinity and $T_{test} = 2.12\ mm$) and $N_{pose} = 7$.

Taking into account Equations (4.4.72) and (4.4.73) in all the simulations we used $F_{out} = 3$ which, see Table 4.7, corresponds to $P_f = 3.12e-3$ and $P_n = 2.26e-2$. Given the size of the pixel in the range image, and the projection of the segment AB

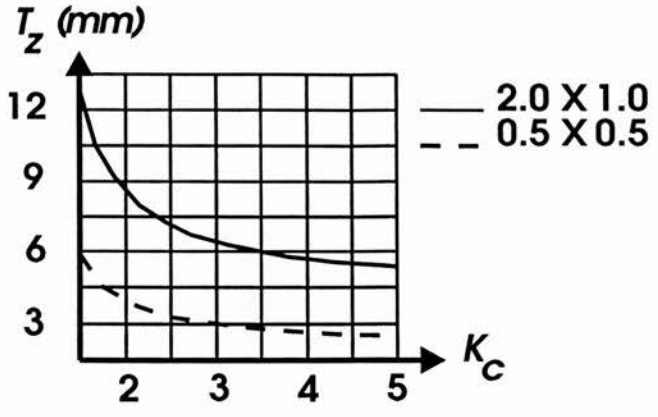


Figure 4.47: Variation of the minimum size of tolerance zone as a function of the value of the process capability index K_c for two pixel sizes: *2 mm* × *1 mm* and *0.5 mm* × *0.5 mm*. Result of four hundred simulations for $N_{pixel} = 10$, $\phi_1 = 0.66$ *rad*, $\phi_2 = 0.51$ *rad*, $\theta = 0.02$ *rad*, $\epsilon = 0.51$ *rad*, $P_f^{feat} = P_n^{feat} = 0.01$ and $\sigma_S = 0.1$ *mm*.

on the image plane, the use of $F_{out} = 3$ means that we will only be able to detect an isolated deformation with size not smaller than *3.0 mm*.

We observed the final diagnosis of the inspection in three different cases:

1. the test tolerance zone center is found by applying a least square procedure to the edge points in F_m (*case 1*),
2. the test tolerance zone center is determined using the *approach 2* in the minimisation (*case 2*) and
3. the test tolerance zone center is determined using the *approach 1* in the minimisation (*case 3*).

In all the simulation runs and for the three different cases, the inspection procedure concluded that the inspected segment was within tolerance and, given the reliability of the process, we had I_p^{shape} equal to $9.774e-1$. Taking into account that the simulated segments were within tolerance and the probability of false alarm of the inspection procedure ($P_f = 3.12e-3$), the expected number of rejections by the inspection procedure was 0 which agrees with the simulation results.

Although the previous results seem to indicate that three ways of positioning the axis of the tolerance zone are equivalent, this fact is not true. Figures 4.48 and 4.49 illustrate

the differences between the three different cases. These figures correspond to the result of 40 simulations, under the same conditions as the previous simulation, but with T_{test} = *1.4 mm* and $F_{out} = 1$.

Figure 4.48 shows the maximum distance between the axis of the tolerance zone and the edge points in F_m in some of the 40 simulation runs. Figure 4.49 shows the number of points out of the tolerance zone in the same simulation runs of Figure 4.48. Both figures show the results of three different ways of positioning the test tolerance zone.

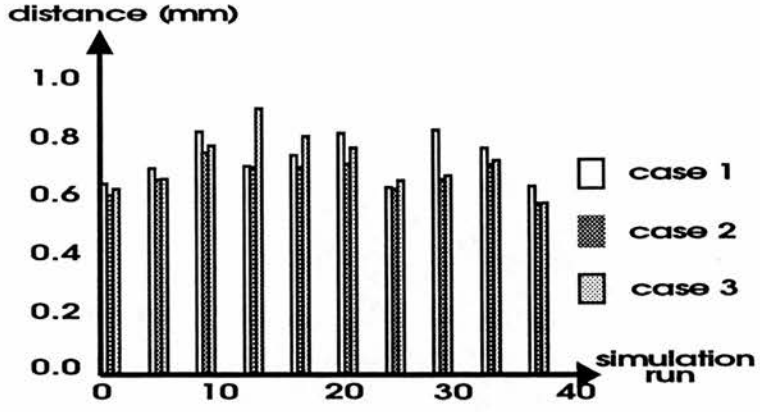


Figure 4.48: Maximum distance between measured edge point and axis of test tolerance zone for 3 different strategies of positioning the test tolerance zone: *case 1* (least square approximation), *case 2* (minimisation of maximum distance) and *case 3* (minimisation of the number of edge points out of tolerance). Result of the runs 1, 5, 9, 13, 17, 21, 25, 29, 33 and 37 of forty simulations for $N_{pixel} = 10$, ϕ_1 = *0.66 rad*, ϕ_2 = *0.51 rad*, θ = *0.02 rad*, ϵ = *0.51 rad*, T_{test} = *1.4 mm*, $N_{pose} = 7$, $F_{out} = 1$, σ_S = *0.1 mm* and pixel size equal to *2 mm* × *1 mm*.

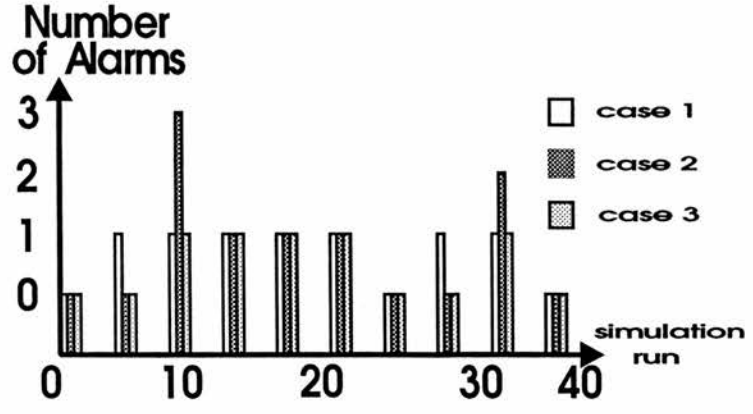


Figure 4.49: Number of points out of the test tolerance zone (number of alarms) for 3 different strategies of positioning the test tolerance zone: *case 1* (least square approximation), *case 2* (minimisation of maximum distance) and *case 3* (minimisation of the number of edge points out of tolerance). Result of the runs 1, 5, 9, 13, 17, 21, 25, 29, 33 and 37 of forty simulations for $N_{pixel} = 10$. Simulation under the same conditions of Figure 4.48.

Looking at Figure 4.48 we observe that there is not a big variation of the maximum distance between the edge points and the axis of the test tolerance zone in the three cases. This fact justifies the common use of *case 1* in many practical situations. However, the comparison of the performances of *case 1* and *case 2* in Figure 4.49 reveals that we might have wrong inspection diagnosis using *case 1*.

Figure 4.49 illustrates how *case 2* and *case 3* are equivalent when just one edge point out of tolerance is enough to conclude that the whole segment is out of tolerance (every time we have zero alarms in *case 2* we also have zero alarms in *case 3*). The figure also illustrates how the performance of the *case 2* is the poorest when we are interested in detecting more than one edge point out of tolerance. This happens because the minimisation of the maximum distance does not imply the minimisation of the number of points out of tolerance.

As one could expect, the results show the best performance of *case 3* for the determination of the pose of the tolerance zone which is comprehensible as it corresponds to the exact implementation of the I_p^{shape} diagnosis.

Figures 4.50 and 4.51 show the result of the straightness inspection applied to the range image of the *widget* shown in Figure 4.44 under the same conditions as the simulation in Figure 4.49 but with $T_{test} = 2.12\ mm$ (K_c equal to infinity and $T_z = 4.2\ mm$).

The figures show the projections of the extracted edge points of the inspected segment (segment AB in Figure 4.46) and of the test tolerance zone on the planes xz and yz of the sensor reference frame. The inspection was carried out using 60 roof edge points extracted using the algorithm of Section 4.3.1.

As in the 200 simulations under the same conditions, the segment was considered within tolerance using all the three different ways of positioning the axis of the test tolerance zone. The maximum distance observed between the axis of the test tolerance zone and an edge point, using *case 2*, was 0.83 mm.

So far, have shown how the ideas discussed in the previous sections can be used in the implementation of straightness inspection procedures under a given set of conditions. These same ideas can be used to determine the appropriate sensor and best viewing conditions for implementing the inspection of a desired segment. As an example, let

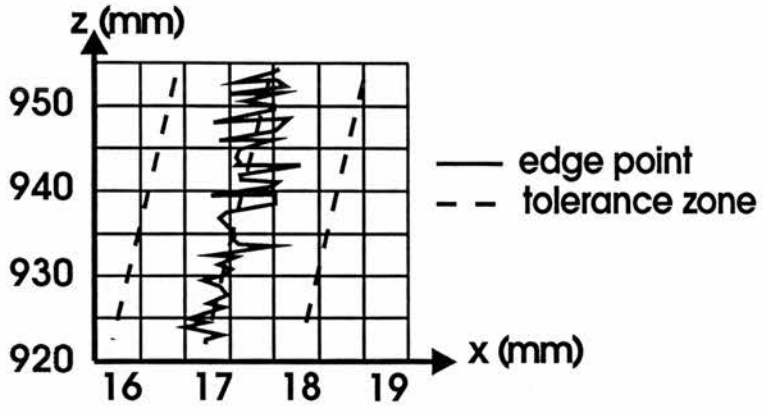


Figure 4.50: Coordinates z and x of extracted edge points and test tolerance zone (positioned using *approach 2*) obtained during the straightness inspection of segment in range image of Figure 4.44 (segment AB in Figure 4.46) for $N_{pixel} = 10$, $T_{test} = 2.12\ mm$, $P_f^{feat} = P_n^{feat} = 0.01$, $F_{out} = 3$, $N_{pose} = 7$.

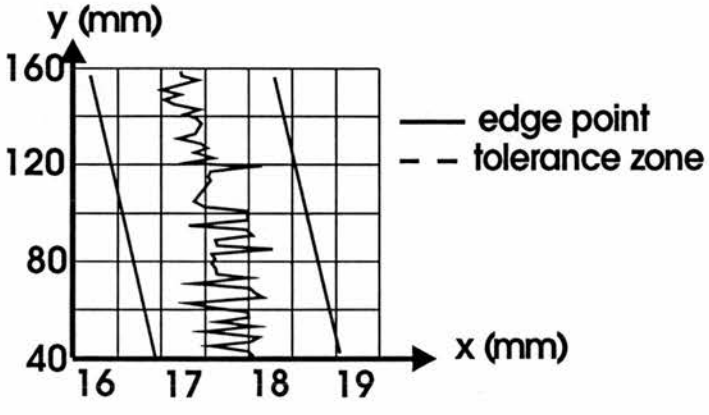


Figure 4.51: Coordinates y and x of extracted edge points and test tolerance zone (positioned using *approach 2*) obtained during the straightness inspection of segment in range image of Figure 4.44 under the same conditions of Figure 4.50.

us consider the straightness inspection of the same segment AB in Figure 4.46 for $P_n^{feat} = P_f^{feat} = 0.01$ again.

For fixed viewing conditions and values of reliability and sensitivity, the accuracy in the extraction of the edge points and, consequently, the minimum size of tolerance zone we can inspect, can only be altered by changing the sensor. This means, changing the pixel size and the value of σ_S.

Figure 4.47 shows the effect of changing the pixel size. The figure shows the variation of the minimum tolerance zone we can measure for the same conditions of Figure 4.50 and $\sigma_S = 0.1\ mm$ for two different sizes of pixel.

ϕ_1 (rad)	ϕ_2 (rad)	θ (rad)	ϵ (rad)	T_z (mm)
0.660	0.510	0.020	0.510	4.20
0.660	0.510	0.000	0.000	1.22
0.585	0.585	0.000	0.000	1.10
0.585	0.585	0.700	0.000	2.36
0.900	0.279	0.000	0.000	1.24

Table 4.8: Minimum value of tolerance zone as a function of the position of the 3D edge line with relation with the sensor reference frame (angles ϕ_1, ϕ_2, θ and ϵ). Results obtained in 200 simulations $P_f^{feat} = P_n^{feat} = 0.01$, $\sigma_S = 0.1$ *mm* and pixel size of *2 mm* $\times$ *1 mm*.

Because of the linearity of the errors with the value of σ_S, the minimum values of tolerance zone corresponding to a value of σ_S different from *0.1 mm* can be obtained by multiplying the values in Figure 4.47 by $10\sigma_S$. For instance, under the same conditions as Figure 4.50 we can inspect a tolerance zone ten times smaller in size when $\sigma_S =$ *0.01 mm*

Table 4.8 shows the variation in the minimum size of tolerance zone in the inspection of the segment AB when we consider different positions of the 3D edge line with respect to the sensor reference frame (angles ϕ_1, ϕ_2, θ and ϵ) for $\sigma_S = 0.1$ *mm* and pixel size equal to *2 mm* $\times$ *1 mm*.

As we can see from the table, the smallest possible tolerance zone we can inspect with the sensor considered is approximately *1.10 mm*. Also, taking into account Figure 4.47, we can expect a reduction of 50 % in the size of tolerance zone if we consider pixels of *0.5 mm* $\times$ *0.5 mm* in size. This means that tolerance zones smaller than approximately *0.6 mm* can only be inspected if we consider sensors with smaller values of σ_S (when the pixel size is not smaller than *0.5 mm* $\times$ *0.5 mm*).

4.6.2 Planarity Inspection

Planarity inspection verifies the shape of a manufactured planar surface. It intends to verify whether the deviations of the shape of the manufactured surface from the shape of an infinite plane are within the desired tolerance or not.

As we are interested in the inspection of the shape of a planar patch we will implement the planarity inspection using I_p^{shape}. Furthermore:

- the set of measured features F_m will be a subset of the range image corresponding to a set of 3D points belonging to the planar patch being inspected,
- the set F_m will be determined using the initial approximation of the projection of the planar patch being inspected on the image plane given by the registration,
- the uncertainty region associated with each 3D point will be a segment of line parallel to the viewing direction corresponding to the error in the range measurement of each pixel in the range image (as before we will model this error as a normally distributed random variable with standard deviation σ_S),
- the test tolerance zone center is an infinite plane whose pose $Pose$ will be defined using a point with coordinates (x_p, y_p, z_p) belonging to the infinite plane and two angles (θ and ϕ) determining the direction of the infinite plane normal.
- the test tolerance zone will correspond to the region between two infinite parallel planes at a distance T_{test} from each other,

The implementation of the I_p shape involves a minimisation process to determine the pose $Pose$ of the test tolerance zone center. In the case of the planarity inspection, the pose $Pose$ is determined by minimising the number of 3D points in F_m at a distance bigger than $\frac{T_{test}}{2}$ from the infinite plane corresponding to the test tolerance zone center (*approach 1*).

Similarly to the case of straightness inspection, it is also possible to consider an alternative strategy that tries to minimise the maximum distance between the tolerance zone center and a 3D point in F_m (*approach 2*).

As in the previous section, the determination of the final pose $Pose$ of the center of the test tolerance zone center was implemented as an exhaustive search in a neighbourhood of the $Pose_m$ ($Pose_m = [x_p^m, y_p^m, z_p^m, \theta^m, \phi^m]$) of the infinite plane corresponding to the least square approximation of the set of measured points F_m (see Equations (4.3.45) to (4.3.50)). The optimal pose is determined by exhaustive search in a discrete grid defined as in Equations (4.6.80) to (4.6.85).

In Section 4.4 we commented on the fact that Equations (4.4.67) and (4.4.69) overestimate the size of the minimum test tolerance zone we can measure. In the case of the

planarity inspection a more accurate estimate of the test of the tolerance zone can be easily obtained. This happens because we have errors only in the z coordinate of the 3D points. Therefore, a variation of size d_{max} on the z coordinate of a 3D point will cause a variation $d_{max}cos(\phi)$ on the distance between the 3D point and the center of the test tolerance zone (where ϕ is the angle between the z axis and the normal of the infinite plane corresponding to the center of the test tolerance zone). This means that in the case of planarity inspection, we can write:

$$T_z = 4cos(\phi)d_i^{max}\frac{K_c}{K_c - 1} \tag{4.6.86}$$

For a probability $P_f^{feat} = P_n^{feat} = 0.0027$ the size of the distance d_{max} of the uncertainty region associated with each 3D point in F_m is equal to $3\sigma_S$. Figure 4.52 shows the corresponding variation of the ratio between the minimum possible value of T_z we can inspect and σ_S as a function of the value of the process capability index K_c (obtained using Equation (4.6.86)). For instance, if $\sigma_S = 0.1\ mm$ the minimum possible value of T_z (corresponding to K_c approaching infinity and $T_{test} = 0.48mm$) is *0.96 mm.*

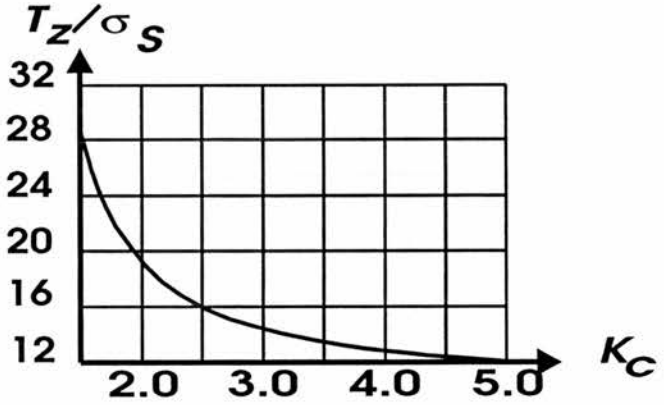


Figure 4.52: Ratio between minimum tolerance zone and sensor standard deviation as a function of the process capability index for $P_f^{feat} = P_n^{feat} = 0.0027$ and $\phi = 0.66\ rad$.

As an example of the planarity inspection let us consider the inspection of a planar patch (patch $ABCD$ in Figure 4.46) in such viewing conditions that:

- the angles ϕ_1 between the normals of the intersecting planes and the viewing direction is equal to *0.66 rad*,

- the plane occupies an area of approximately *60 mm* × *80 mm* on the image plane and has a hole in its middle (see Figures 4.44 and 4.46)

Given the pixel size of the range image, the total number of pixels in F_m is 2400. This means that if we assume $F_{out} = 14$ and $T_{out} = 18$ the values of P_f and P_n according to Equations (4.4.72) and (4.4.73) will be respectively equal to $2.8e-3$ and $2.0e-3$ (for $P_f^{feat} = P_n^{feat} = 0.0027$).

The use of $T_{out} = 18$ means that we will only be able to detect, with probability not smaller than 0.998, deformations covering regions of size not smaller than approximately *6 mm* × *6mm* considering a pixel size of *2 mm* × *1 mm*.

In order to observe the performance of the planarity inspection, we proceeded with 100 simulations in which we inspected synthetically created images of the plane described above. The synthetic images were created considering $\sigma_S = 0.1$ *mm* and pixel size *2 mm* × *1 mm*. Also, we used $T_{test} = 0.48$ *mm* (corresponding to K_c approaching infinity and $P_f^{feat} = P_n^{feat} = 0.027$), $F_{out} = 14$ and $N_{pose} = 7$.

As before, we considered the 3 different ways of positioning the test tolerance zone described in the previous section: *case 1*, *case 2* and *case 3*.

As one could expect, given the number of simulations and the values of P_f and P_n, the synthetically created planes were considered within tolerance in all the simulations ($I_p^{shape} = 0.998$).

Although this result seem to indicate that the minimisation process used to determine the *Pose* of the tolerance zone is not necessary (the least square approximation was enough in all the simulations), the differences in performance of the 3 different ways of positioning the test tolerance zone are basically the same as we observed in the case of the straightness inspection.

Similar results to the ones shown in Figure 4.50 and Figure 4.51 can be obtained under the same simulation conditions, but for smaller values of T_{test}.

As a final example let us apply the planarity inspection to the same planar patch ($ABCD$ in Figure 4.46) and under the same conditions of the simulation, but this time using the actual range image of the planar patch in Figure 4.44 with $T_{test} = 0.48$

mm, $N_{pose} = 11$ and an approximation of the inner points of the patch supplied by hand.

Under these conditions, the surface was rejected using the three different ways of positioning the test tolerance zone and, given the value of P_f, we obtained $I_p^{shape} = 0.0027$.

Also, the maximum distance between a point on the manufactured surface and the center of the test tolerance zone was equal to *1.5 mm* when the surface was positioned according to *case 1*, and *1.1 mm* when *case 2* was used. These numbers indicate that the manufactured surface presents deviations of almost two times the size of the tolerance zone from the shape of an infinite plane. The numbers also show that the manufactured surface could only be considered within tolerance for values of K_c smaller than approximately *1.5* (see Figure 4.52).

The application of the planarity inspection to the planar manufactured surface in Figure 4.44 corresponding to the plane $ABEGF$ in Figure 4.46 (under the same conditions of the inspection of plane $ABCD$) also rejected the manufactured surface using the three different ways of positioning the test tolerance zone. This fact suggests that the final quality in the shape of the planar surfaces of the widget in Figure 4.44 is very poor or our inspection procedure is too demanding.

The determination the appropriate sensor to inspect the planarity of a given planar patch can be easily done using Equation (4.6.86). The application of this Equation for the case in which $P_f^{feat} = P_n^{feat} = 0.0027$ and $\phi = 0.66$ *rad* is illustrated in Figure 4.52. For instance, we can conclude from the figure that the inspection of a tolerance zone of size *0.5 mm* for $K_c = 2.5$ is only possible if we have a sensor with σ_S smaller than *0.03 mm*.

According to Equation (4.6.86) the minimum size of tolerance zone that we can measure approaches 0 when ϕ approaches 90°. However we cannot forget that when ϕ approaches 90°, the plane becomes less and less visible in the range image. This means that we should avoid angles near of 90° in order to guarantee a good visibility of the plane. In the ideal inspection conditions we would like to have ϕ equal to 0° and a sensor with a value of σ_S appropriate to measure a desired size of tolerance zone with

the desired values of reliability and sensitivity.

4.6.3 Hole shape inspection

We will use the term hole shape inspection to designate the inspection of the shape of the curve created by the intersection of a manufactured planar surface and a manufactured circular cylindrical surface. As in Section 4.3.3 we will only consider cases in which the viewing direction is parallel to the normal of the planar surface and to the axis of the cylindrical surface. Our final objective is the inspection of the circularity of circular cylindrical holes in manufactured objects.

As the hole shape inspection is concerned with the shape of a manufactured circular curve, we will implement the I_p^{shape} diagnosis and:

- the set F_m of measured features it will be a set of 3D step edge points extracted using the algorithm described in Section 4.3.3,
- the test tolerance zone center will be a circle with pose $Pose$ defined by the coordinate x_{center} and y_{center} of the center of the circle and size $Size$ defined by the radius R of the circle and
- the test tolerance zone will be a subset of the plane xy delimited by two circles of center x_{center} and y_{center} and radius respectively equal to $R + \frac{T_{test}}{2}$ and $R - \frac{T_{test}}{2}$.

We will determine the pose and the size of the center of the test tolerance zone using an exhaustive search in a discrete grid with ${N_{pose}}^3$ elements in which the element (i, j, k) ($[x^i_{center}, y^j_{center}, R^k]$) will be defined as:

$$x^i_{center} = x^M_{center} - \frac{N_{pose}}{2}\Delta_{loc} + i\Delta_{loc} \tag{4.6.87}$$

$$y^j_{center} = y^M_{center} - \frac{N_{pose}}{2}\Delta_{loc} + j\Delta_{loc} \tag{4.6.88}$$

$$R^k = R^M - \frac{N_{pose}}{2}\Delta_{loc} + k\Delta_{loc} \tag{4.6.89}$$

where:

- Δ_{loc} is the same of Equations (4.6.80) to (4.6.85) and

- x^M_{center}, y^M_{center} and R^M are respectively the center coordinates and the radius of the circle obtained during the extraction of the edge points in F_m - see Section 4.3.3.

In order to exemplify the implementation of the I_p^{shape} diagnosis, let us consider the inspection of the hole H in Figure 4.46 : radius equal to *10 mm*, center at the coordinates $x = 140$ *mm* and $y = 60$ *mm* and a C_0 discontinuity of *300 mm*. Let us also assume that we are using a laser striper with $\sigma_S = 0.1$ *mm* and pixels of size *0.28 mm* $\times$ *0.30 mm*.

Under these conditions, according to the results of 100 simulations, the value of the maximum absolute error in the estimate of the x and y coordinates of the edge points of the hole border is *0.90 mm*. Therefore, following the ideas in Section 4.3.3, we will assume that for a probability $P = 0.01$ the uncertainty regions associated with the measured edge points are such that $d_{max} = 0.90$ *mm*. Figure 4.53 shows the variation of the minimum size of tolerance zone we can inspect for this value of d_{max} as a function of the process capability index K_c.

For $d_{max} = 0.90$ *mm* the minimum possible size of tolerance zone we can inspect according to Equation (4.4.67) is *3.6 mm* which corresponds to $T_{test} = 1.8$ *mm*.

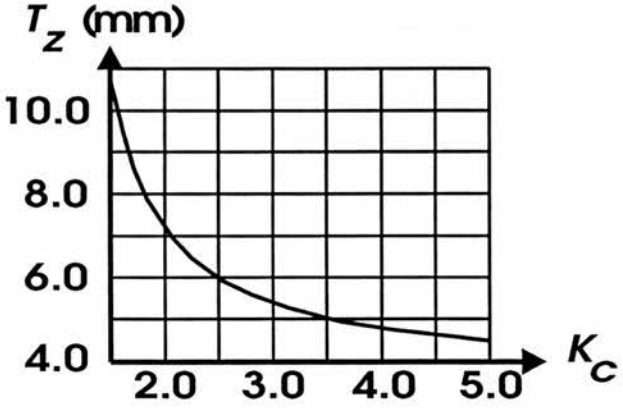


Figure 4.53: Variation of the minimum size of tolerance zone possible to inspect with the process capability index $K_c for\ P_f = P_n = 0.01$, pixel size equal to *0.28 mm* $\times$ *0.30 mm* and $\sigma_S = 0.1$ *mm*.

To observe the performance of the hole shape inspection procedure, we applied the inspection procedure to 200 synthetically created images of holes under the same conditions discussed so far and with $T_{test} = 1.8$ *mm*. In all the simulations the relative

position of the hole with respect to the pixels of the range image was generated randomly and we always used $N_{pose} = 7$, $F_{out} = 5$ and inspected 180 edge points.

According to Equations (4.4.72) and (4.4.73), assuming $T_{out} = 7$, these conditions correspond to $P_f = 1.0e-3$ and $P_n = 3.7e-4$. Furthermore, the assumption of T_{out} equal to 7 implies that we are only concerned with isolated deformations corresponding to a length of at least approximately *2.4 mm* or an arc of at least 14°.

As in the two previous sections we considered the 3 different ways of positioning the center of the test tolerance zone: *case 1*, *case 2* and *case 3* of Section 4.6.1 .

During the 200 simulations the surface was always accepted for the 3 ways of positioning the center of the test tolerance zone ($I_p^{shape} = 0.9996$) which agrees with the expected number of false alarms in the experiment (zero).

The differences among the three ways of positioning the center of the test tolerance zone observed during the simulations are basically the same ones that were observed in the previous implementations of the I_p^{shape}. The differences can be easily observed by reducing the values of F_{out} or T_{test}. For instance, under the same conditions, but with F_{out} equal to one, we observed 50 rejections when the center of the test tolerance zone was positioned according to *case 1*.

The application of the hole inspection procedure to the range image of hole H in Figure 4.45 demanded an alteration in the algorithm of Section 4.3.3. The alteration was necessary because (see Figures 4.45 and Figure 4.54) spurious reflections of the laser on the interior surface of the hole originated values of range measurement completely wrong. This problem can be clearly seen in Figure 4.54 which shows the profile of hole H in a row of the range image.

In order to solve the problem caused by the spurious reflections of the laser striper we altered the initial part of the algorithm in which the inner region of the hole was determined by thresholding. This part of the algorithm, that we will call *cleaning*, was changed to:

1. Determine average value of height outside the hole (z_{AV}) and used this value to determine a threshold value (normally equal to $Z_{AV} - 10\sigma_S$).

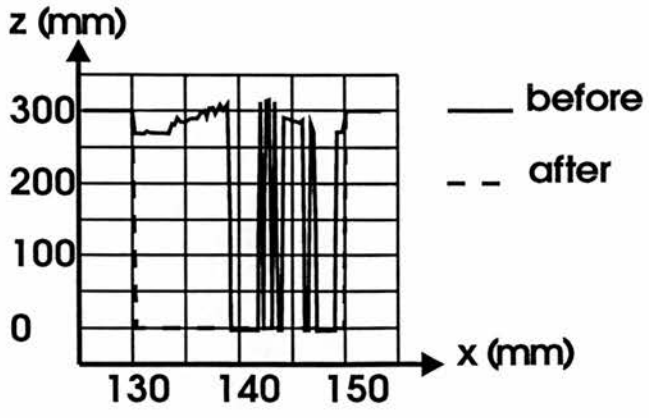


Figure 4.54: Profile of hole in Figure 4.45 before and after elimination of spurious reflections. Profile corresponding to row 50 in range image.

2. Determine the first approximation of the inner region of the hole by processing each row of the range image in the following way:

   (a) Starting from the outside of the hole and from the hole's right find the first pixel whose range measurement is smaller than threshold value (we will call this point x_{right}).

   (b) Starting from the outside of the hole and from the hole's left find the first pixel whose range measurement is smaller than the threshold value (we will call this point x_{left}).

   (c) Set all the pixels in the row between x_{right} and x_{left} to the value $z_{AV} - step$ (where *step* is the depth of the hole). All the pixels between x_{right} and x_{left} are in the approximation of the inner region.

3. Find the second approximation of the inner region of the hole by processing each column of the range image as described above.

4. Find the final approximation of the inner region of the hole by the union of the first and second approximation of the inner region of the hole.

Figure 4.54 illustrates the result of the *cleaning* applied to the Figure 4.45. The figure shows the hole profile in a row of the range image before and after the elimination of the reflections. The figure shows a typical performance of the *cleaning* that was always very effective in all our experiments with range images of holes.

After the *cleaning* we proceeded the extraction of the step edge points exactly as described in Section 4.3.3. Figure 4.55 shows the extracted edge points, as well as the projection of the test tolerance zone on the range image. This figure was the result of applying the hole shape inspection diagnosis to the range image for $T_{test} = 1.8$ mm using $N_{pose} = 9$ and using 23 pixels in the determination of each edge point. The inspection accepted the hole using the 3 different ways of positioning the center of the test tolerance zone ($I_p^{shape} = 0.9996$).

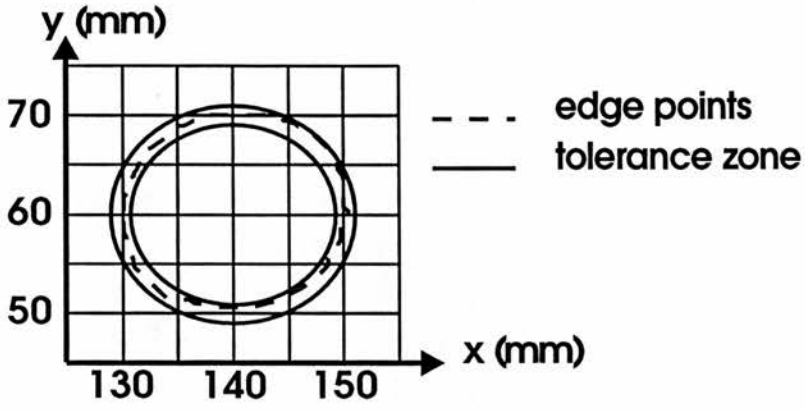


Figure 4.55: Extracted edge points and tolerance zone in inspection of hole in Figure 4.45. Result of inspection with $N_{pose} = 9$ and step edge points extracted using 23 pixels.

The application of the inspection procedure to the range image also showed a good illustration of the differences in performance between the different ways of positioning the test tolerance zone. For instance, with $F_{out} = 1$ and T_{test} *1.6 mm* the hole was only accepted using *case 2* or *case 3* (for N_{pose} at least equal to 29).

The determination of the appropriate sensor to accomplish a desired hole shape inspection task is complicated by the use of the *cleaning* operation in many cases. However, in our experiments the effects of the *cleaning* in the final accuracy of the extraction of the edge points was negligible.

This means that the determination of the appropriate sensor can still be done using Monte Carlo simulations (in which we do not take the spurious reflections into account) and Equation (4.4.67). Therefore, Figure 4.53 can be used to determine the minimum possible size of tolerance zone that is possible to inspect as a function of the process capability index.

Despite the fact that Figure 4.53 was obtained for $\sigma_S = 0.1\ mm$, the figure is also valid for other values of σ_S (between $0.01\ mm$ and $1\ mm$). This happens because the errors in the extraction of the edge points are much more related with the size of the image pixels (and the relative position between the hole's center and the pixels in the range image) than with the value of σ_S.

As we cannot find laser stripers with pixels of size much smaller than $0.28\ mm \times 0.3\ mm$ and the errors in the extraction do not vary much with the radius of the hole, Figure 4.53 is a good illustration of the limits in the inspection of the shape of holes using range data and the algorithm of Section 4.3.3.

4.6.4 Angularity inspection - angle between two lines

In this section we will be concerned with the inspection of the angle between two lines when each of these lines is produced by the intersection of two manufactured planar surfaces.

Although the ANSI standards defines two particular cases in which the angle is either 90° (perpendicularity) or 0° (parallelism), we will not make such a distinction and we will use the term angularity inspection to designate the inspection of any angle varying from 0° to 180°.

We will implement the angularity inspection procedure using the algorithm described in Section 4.3.6 for the measurement of the angle between two segments of line.

It is important to observe that our implementation of the angularity inspection does not follow the tolerance standards. We opted for doing so, because the angularity inspection according to the standards corresponds to the application of the I_p^{pose} diagnosis in which one of the intersecting lines is considered as a datum (see Section 4.6.9 for an illustration of an implementation of the I_p^{pose} diagnosis).

In order to illustrate the application of the angularity inspection procedure let us consider its application to verify the angle between two perpendicular segments in Figure 4.44 corresponding to the segments AB (the intersection between the planes $ABCD$ and $ABEGF$) and AF (the intersection between the planes $AFIJD$ and $ABEGF$) in Figure 4.46.

The viewing conditions of these two segments in Figure 4.44 can be summarised by:

- the segments AB and AF project on the image plane as shown in Figure 4.46,
- the planes $ABCD$, $ABEGF$ and $AFJID$ make respectively angles equal to *0.66 rad*, *0.51 rad* and *1.06 rad* with the viewing direction,
- the segments AB and AF make respectively angles of approximately *0.51 rad* and *0.0 rad* with the image plane.

Figure 4.56 shows the variation of the minimum size of tolerance zone that is possible to inspect as a function of the process capability index for $P_f = P_n = 0.005$ and different values of N_p (number of pixels used to calculate the least square approximation of the planes $ABCD$, $ABEFG$ and $AFJID$). The figure was obtained by using Equation (4.4.67) and the procedure described in Section 4.3.6 for determining the uncertainty region in the estimation of the angle α between two segments (200 simulations were considered).

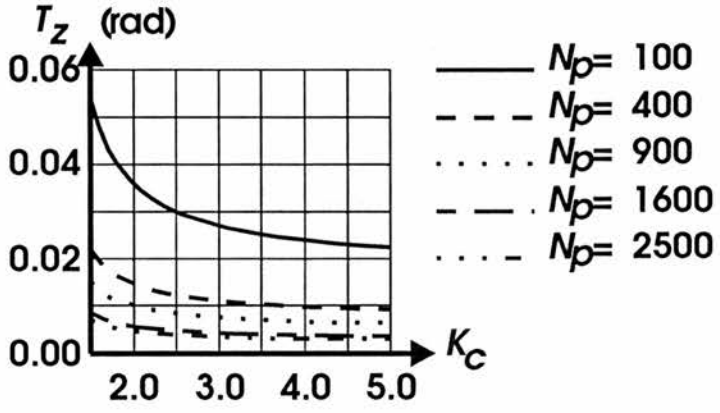


Figure 4.56: Minimum size of tolerance zone in angularity inspection of segments AB and AF for $P_f = P_n = 0.01$. Result obtained in 200 simulations for five different values of N_p: 100, 400, 900, 1600 and 2500. Laser striper assumed to have $\sigma_S = 0.1$ *mm*.

In the application of the angularity inspection procedure to the segments AB and AF, the least square approximation of the planes $ABCD$, $ABEF$ and $AFJID$ was calculated using the pixels belonging to these planes in the range image of Figure 4.44 that were supplied by hand - see Figure 4.46. As we can see in Figure 4.46 we considered rectangular regions with one of the sides parallel to the segments being inspected and the other side with size equal to approximately ten pixels. This means

N_p	d_{max} (rad)
100	5.5e-3
400	1.83e-3
900	1.25e-3
1600	7.1e-4
2500	5.8e-4

Table 4.9: Size of uncertainty region (d_{max}) associated with the measurement of the angle between the segments AB and AF as a function of N_p for a probability $P = 0.005$. Result of 200 simulations to each value of N_p. Sensor assumed to have $\sigma_S = 0.1\ mm$.

that the number of pixels N_p used in the least square approximation of the planes varied between 400 and 800.

As the result of the application of the angularity inspection to the range image of Figure 4.44 we obtained $\alpha = 1.535\ rad$ which corresponds to a deviation of $0.035\ rad$ from the nominal value ($1.57\ rad$). Taking into account that we used N_p between 400 and 800, according to Equation (4.4.67) and Table 4.9 for $P_f = P_n = 0.005$ the angle would be considered within tolerance only for values of K_c smaller than approximately 1.1.

Table 4.9 and Equation (4.4.67) can also be used for determining the appropriate sensor to accomplish a given angularity inspection. As an example, let us consider the inspection of the segment AB and AF again under the same viewing conditions and the same portions of the planes $ABCD$, $ABEF$ and $AFJID$ with $P_n = P_f = 0.005$.

Under these conditions, the effect of changing the pixel size can be observed in Figure 4.56 as the use of pixels of smaller size allows the use of bigger values of N_p. For instance, the use of pixels of size $1\ mm \times 1\ mm$ would allow the use of N_p between 800 and 1600 and consequently we would be able to inspect tolerance zones of size $0.01\ rad$ for values of K_c bigger than 2 (contrarily to the case of Figure 4.44 in which this is possible only for values of K_c bigger than 4).

The effect of changing the value of σ_S can be estimated considering the linearity of the errors with the variation of the value of σ_S. For instance, according to Figure 4.56 for N_p equal to 400 and K_c equal to 2 we would be able inspect tolerance zones of size $0.0015\ rad$ using a value of σ_S ten times smaller ($\sigma_S = 0.01\ mm$).

4.6.5 Angularity inspection - angle between two planes

In this section we will be concerned with the inspection of the angle between two manufactured planar surfaces. As in the previous section we will use the term angularity inspection to designate the verification of any angle from 0° to 180°.

We will implement the angularity inspection using the algorithm described in Section 4.3.5 for measuring the angle between the two manufactured surfaces we are interested to inspect.

As in the previous section, our implementation of the angularity inspection does not follow the tolerance standards. We opted for doing so, because the angularity inspection according to the standards corresponds to the application of the I_p^{pose} diagnosis in which one of the intersecting planes is considered as a datum.

As an example, let us consider the inspection of the angle between the planes $ABCD$ and $ABEGF$ in Figure 4.46 using a laser striper with $\sigma_S = 0.1\ mm$ and pixels of size $2\ mm \times 1\ mm$.

In order to determine the size of the uncertainty region in the determination of the angle α between the two planes (for a probability $P = 0.01$) we proceeded as described in Section 4.3.5 and observed the result of 200 simulations for three different values of N_p: 100, 400 and 1600.

As a result of the simulations we concluded that the values of d_{max} associated with each value of N_p were respectively equal to *2.70 e-3 rad*, *6.70 e-4 rad* and *3.75 e-4 rad*. The corresponding values of minimum size of tolerance zone, obtained using Equation (4.4.67), are shown in Figure 4.57 as a function of the process capability index.

By applying the algorithm described in Section 4.3.5 to the range image in Figure 4.44 with N_p equal to 1600 we obtained α equal to *0.46353 rad*. Table 4.10 shows the different values obtained for α using N_p equal to 100 and 400, when we consider different subsets of the range image to estimate the normal of the two planes.

The table also shows the maximum value of process capability index K_c for which each measured value of α would be considered within tolerance ($I_p^{global} = 0.99$). The maximum value of K_c associated with each measurement of angle in Table 4.10 was

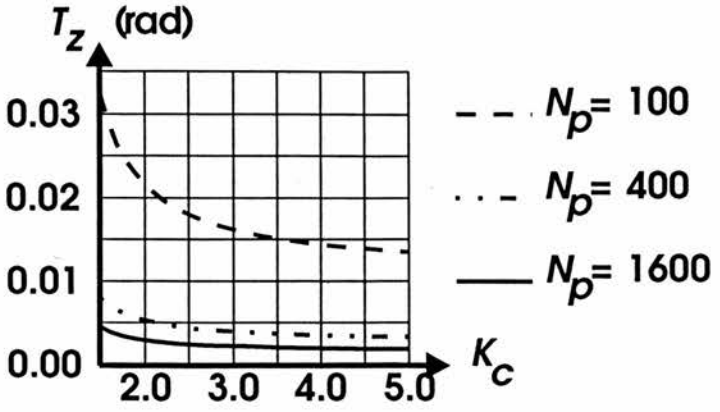


Figure 4.57: Minimum value of tolerance zone in angularity inspection of planes $ABCD$ and $ABEGF$ for $P_f = P_n = 0.01$. Result obtained in 200 simulations for three different values of N_p: 100, 400 and 1600. Laser striper assumed to have pixels of size *2 mm* × *1 mm* and σ_S = *0.1 mm*.

$N_p = 100$		$N_p = 400$	
ANGLE (rad)	MAX K_c	ANGLE (rad)	MAX K_c
0.473693	1.72	0.467672	1.39
0.465629	∞	0.463459	∞
0.458438	3.26	0.461757	2.21
0.474873	1.62	0.469080	1.27
0.469769	2.53	0.465964	1.76

Table 4.10: Values of the measured angle α and corresponding maximum value of process capability index K_c for which each measured angle α would be considered within tolerance. Result of angularity inspection applied to the surfaces in Figure 4.44 corresponding to planes $ABCD$ and $ABEGF$ in Figure 4.46. Angularity inspection done using two different values of N_p (100 and 400), considering different portions of the planes $ABCD$ and $ABEGF$ and assuming the nominal value of angle equal to *0.46353 rad*.

calculated assuming that the nominal value of α was equal to *0.46353 rad* (that is the measured value of α for $N_p = 1600$) and using Equation (4.4.67).

The big variation in the results shown in Table 4.10 can be understood if we consider that the two planar surfaces whose angle we want to measure were rejected in the planarity inspection (we observed deviations up to *1.1 mm*). Therefore, different portions of the manufactured surfaces will correspond to different infinite planar surfaces which explains the variation in the measured angle α.

The determination of the appropriate sensor to accomplish a desired angularity task can be easily done using simulations and Equation (4.4.67).

Figure 4.57 illustrates how it is possible to use simulations and Equation (4.4.67) for determining the appropriate sensor to measure the angle between two given planes. The figure shows the minimum value of T_z that is possible to inspect as a function of the value of the process capability index in the inspection of the planes $ABCD$ and $ABEGF$.

Due to the linearity of the errors with σ_S the figure can also be used to determine the minimum value of tolerance zone we can inspect using different sensors. For instance, tolerance zones of *2 e-3 rad* with $K_c = 4$ can only be inspected by using N_p bigger than 1600 or using a sensor σ_S smaller than *0.05 mm*.

4.6.6 Segment size inspection

In this section we will be concerned with the inspection of the length of the curve resulting from the intersection of two manufactured planar surfaces. We will only consider the cases in which both planar surfaces are visible in the range image. Furthermore, we will consider two different approaches in the inspection of the segment:

- the conventional approach in which we measured the length of the segment and
- according to the size tolerance zone defined in Requicha's theory.

For the implementation of the conventional approach we will use the algorithm described in Section 4.3.4 to determine the length of the segment being inspected. As an example let us consider the inspection of the size of the segment in Figure 4.44 that corresponds to segment AB in Figure 4.46. Let us also assume that besides the segment AB we will use the segments AF (roof edge in Figure 4.44) and segment BE (step edge in Figure 4.44) to determine the vertices A and B - see Figure 4.46.

We will model the errors in the extraction of the edge points of segments AB and AF using the procedure described in Section 4.3.1.

Under the same viewing conditions of Figure 4.44 and for a sensor with $\sigma_S = 0.1$ *mm* and pixels of size *2 mm* $\times$ *1 mm*, according to the result of 200 simulations for each segment, the errors in the determination of the edge points in segment AB and

AF can be modelled as a normally distributed 2D random vector with mean zero, standard deviation *0.40 mm* on the x direction and standard deviation *0.15 mm* on the y direction. Furthermore the errors in the x and y direction can be considered uncorrelated.

Using the normal model of the errors in the extraction of the roof edges and the simulation procedure described in Section 4.3.4 we concluded that the uncertainty region associated with the length measurement, for a probability $P = 0.01$, has a value of d_{max} equal to *0.88 mm*.

This result was obtained as a result of 200 simulations in which we considered 65 edge points in segment AB, 20 edge points in segment AF and 10 edge points in segment BF. Figure 4.58 shows the corresponding variation of the minimum size of tolerance zone that we can inspect as a function of the process capability index according to Equation (4.4.67).

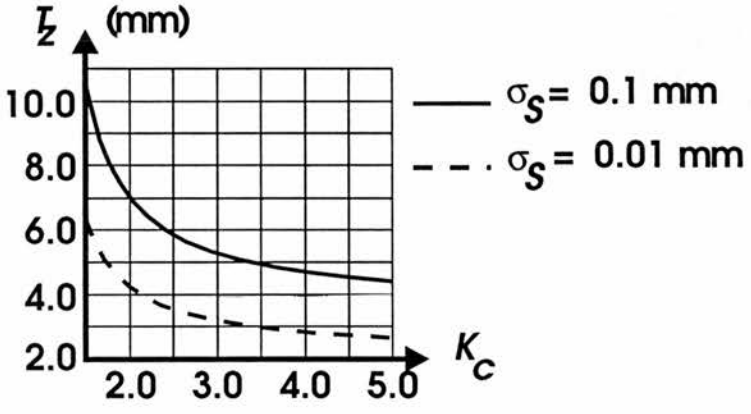


Figure 4.58: Minimum value of tolerance zone in size inspection for $P_f^{feat} = P_n^{feat} = 0.01$ as a function of the process capability index (K_c). Result obtained in 200 simulations under the same viewing conditions of Figure 4.44, 65 edge points in segment AB, 20 edge points in segment AF and 10 edge points in segment BE. Laser striper assumed to have pixels of size *2 mm* × *1 mm*. Result for two different values of σ_S: *0.1 mm* and *0.01 mm*.

The application of the conventional inspection procedure to the segment AB in Figure 4.44 produced a measurement of length equal to *90.26 mm*. This means, that for a nominal value of length equal to *90.0 mm* the segment would be considered within tolerance for any value of process capability index.

The implementation of the size inspection according to Requicha's ideas correspond to the implementation of the I_p^{size} diagnosis described in Section 4.5. We implemented the

I_p^{size} diagnosis in the same way we implemented the straightness inspection in Section 4.6.1. The only differences between the two implementations were:

- the set of measured features F_m in the case of the size inspection is composed not only by the roof edge points extracted according to the algorithm described in Section 4.3.1, but also by the two vertices of the segment that were extracted using the algorithm of Section 4.3.4.
- the center of the test tolerance zone in the case of the size inspection is a segment and not an infinite straight line as in the case of the straightness inspection,
- the pose *Pose* of the center of the test tolerance zone was determined using one vertex of the segment and two angles defining the direction of the segment.

According to the result of 200 simulations under the same viewing conditions of Figure 4.44, assuming a sensor with $\sigma_S = 0.1\ mm$ and pixel size of $2\ mm \times 1\ mm$ the sizes of the uncertainty regions associated with the extraction of the vertices A and B are always smaller than the size of the uncertainty region in the extraction of the roof edge points of segment AB (for the same probability P).

As a consequence of this fact, the graph in Figure 4.47 showing the variation of the minimum size of tolerance zone that we can inspect as a function of the process capability index is still valid. Also, the minimum size of tolerance zone we can inspect (corresponding to K_c approaching infinity) is equal to $1.8\ mm$.

As in the case of the straightness inspection we proceeded 200 simulations in which we observed the result of the inspection procedure for the 3 different ways of positioning the center of the test tolerance zone described in Section 4.6.1 (*case 1*, *case 2* and *case 3*).

The segment was considered within tolerance in all the simulations and for all the three different ways of positioning the center of the test tolerance zone. Furthermore, we also observed the same differences in performance between *case 1*, *case 2* and *case 3* that we observed in the straightness inspection in other simulations of the size inspection.

The segment AB was also considered within tolerance ($I_p^{size} = 0.9996$) when we applied the inspection procedure to the actual range image of Figure 4.44. As in the simulation, the inspection accepted the segment for the three different ways of positioning the test tolerance zone.

The determination of the appropriate sensor for accomplishing a given inspection task can be done using the same procedure used in the case of the straightness inspection. However, it is important to remember that the errors in the extraction of the vertices will not vary linearly with the value of σ_S when one of the segments used in the extraction of the vertices is a step edge.

For instance, in the case of segment AB the errors in the extraction of vertex B do not vary linearly with the value of σ_S. Because of this fact the graphs in Figure 4.47 can only be used if σ_S is equal to *0.1 mm*.

In the case of the conventional size inspection the appropriate sensor can also be determined by using Monte Carlo simulations and Equation (4.4.67).

Figure 4.58 illustrates this procedure in the case of segment AB. The figure shows the variation of the minimum size of tolerance zone we can inspect with $P_f^{feat} = P_n^{feat} = 0.01$ for two different values of σ_S and pixels of size *2 mm* $\times$ *1 mm*. The values of tolerance zone associated with pixels of size *0.5 mm* $\times$ 0.5 mm under the same conditions are approximately 30 % smaller than the ones shown in Figure 4.58.

A comparison between the application of the two approaches to the inspection of the size of the segment AB reveals that, for the same measured features, we were able to achieve a better reliability and sensitivity in the case of the inspection according to Requicha's ideas.

This difference can be explained by two factors: the length measurement is based in the value of only two features (the segment vertices) and the errors in the determination of the vertices are of the same order of magnitude as the errors in the determination of the edge points in segment AB.

4.6.7 Hole size inspection

This section demonstrates the inspection of the size of the curve created by the intersection of a manufactured planar surface and a manufactured circular cylindrical surface. As in Section 4.6.3 we will only consider the cases in which the axis of the cylindrical surface and the normal of the planar surface are both parallel to the viewing direction.

The hole size inspection can be done according to two different approaches:

1. the conventional size inspection in which we measure the radius of the hole's border and
2. according to the size tolerance zone defined in Requicha's theory.

We will implement the conventional inspection of the hole size using the algorithm of Section 4.3.3 to extract the set F_m of edge points corresponding to the hole's border and then using Equation (4.3.26) to calculate the value of the radius.

As an example let us consider the inspection of the hole H in Figure 4.46. If we consider a sensor with $\sigma_S = 0.1\ mm$ and pixels of size $0.28\ mm \times 0.3\ mm$, according to the result of 100 simulations the maximum error in the estimate of the radius is equal to $0.05\ mm$.

Following the ideas in Section 4.3.3, we will assume that for a probability $P = 0.01$ the uncertainty region associated with the determination of the radius (that is a segment) has $d_{max} = 0.05\ mm$. Figure 4.59 shows the variation of the minimum size of tolerance zone we can inspect as a function of the capability constant in this case.

The application of the traditional hole size inspection to the range image of hole H in Figure 4.45 was only possible after the elimination of the errors caused by the spurious reflections of the laser striper using the *cleaning* operation described in Section 4.6.3.

After the cleaning, the conventional inspection procedure measured the radius of the hole equal to $10.0038\ mm$. If we assume that the nominal value of the radius H is equal to $10\ mm$, this result means that the hole can be considered within tolerance ($I_p^{global} = 0.99$) for any value of K_c.

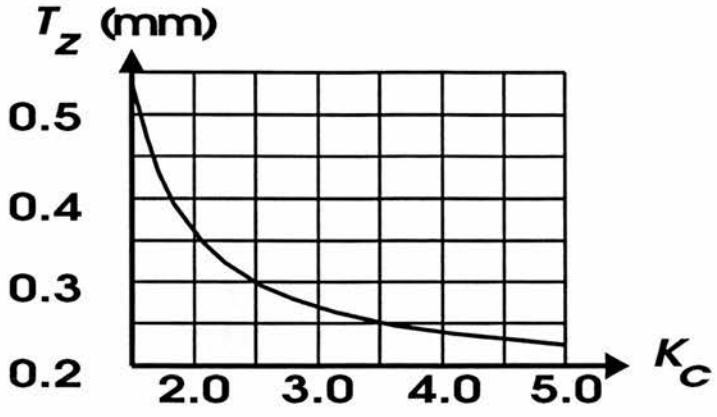


Figure 4.59: Minimum value of tolerance zone in the conventional hole size inspection of hole H in Figure 4.46 for $P_f = P_n = 0.01$ as a function of the process capability index K_c. Result obtained in 100 simulations for $\sigma_S = 0.1\ mm$ and pixels of size $0.28\ mm \times 0.3\ mm$.

The implementation of the hole size inspection according to Requicha's theory of tolerance correspond to the implementation of the I_p^{size} diagnosis described in Section 4.5 and it is very similar to the hole shape inspection described in Section 4.6.3.

The only difference between the two cases is that in the size inspection the radius of the test tolerance zone center is fixed and equal to the nominal radius of the hole. This means that the hole size inspection involves only the determination of the pose of the center of the test tolerance zone (x_{center} and y_{center}) that minimises the number of points outside the test tolerance zone.

We implemented the hole size inspection using the same strategy we used in the implementation of the hole shape inspection, but applying the exhaustive search just to a discrete grid of values of x_{center} and y_{center} with ${N_{pose}}^2$ elements. This two dimensional search grid was defined using the same values of x_{center} and y_{center} shown in Equations (4.6.87) and (4.6.88).

In order to exemplify the implementation of the I_p^{size} diagnosis, let us consider again the inspection of the hole H of Figure 4.46 with $P_f^{feat} = P_n^{feat} = 0.01$, $P_f = 1.0e-3$, $P_n = 3.7e-4$ and $F_{out} = 5$, as in Section 4.6.3. Under these conditions the variation of the minimum size of tolerance zone we can inspect as a function of the process capability index is the same as shown in the Figure 4.53 of Section 4.6.3.

We observed the performance of the hole size inspection procedure in 200 simulations in which we assumed $T_{test} = 1.8\ mm$ and the nominal radius of the hole equal to

10 mm. As before, we also observed the results of using the three different ways of positioning the test tolerance zone center (*case 1*, *case 2* and *case 3* in Section 4.6.1) and $N_p = 7$.

In the 200 hundred runs of simulation the hole was never considered out of tolerance ($I_p^{size} = 0.9996$) for any of the three ways of positioning the center of the test tolerance zone which agrees with our initial expectations. Also, during other simulations we observed that the difference in performance between the 3 different ways of positioning the test tolerance zone is basically the same as Section 4.6.3.

The inspection of the size of the hole H in the range image of Figure 4.45 according to Requicha's ideas produced the same results observed in the shape inspection of the hole in Section 4.6.3. As in the Section 4.6.3 the hole was considered within tolerance for a tolerance zone of size $T_{test} = 1.8\ mm$ using the three different ways of positioning the center of the test tolerance zone. Also, as in Section 4.6.3, we had to apply the *cleaning* operation to the range image.

The comparison between the implementations of the conventional size inspection and the size inspection according to Requicha's ideas in the case of hole H of Figure 4.46 reveals a fundamental difference in the accuracy of the two approaches. As we could expect, the conventional inspection of size is able to check tolerance zones of much smaller sizes. This happens because the uncertainty in the estimate of the hole's radius depends upon a global property of the measured edge points (see Equation (4.3.26)) and it is, therefore, smaller than the uncertainty in a unique measured edge point.

Another important difference is the fact that the inspection according to Requicha's theory is clearly more demanding than the conventional size inspection: the same set of measured features that produced an error in the radius measurement of *0.0038 mm* during the inspection of the hole in Figure 4.45, could only be put inside a tolerance test zone of size at least equal to *1.8 mm.* This difference happened because of an asymmetry in the measured edge points that conventional size inspection was not able to detect - see Figure 4.55.

The determination of the appropriate sensor to carry out a desired hole size inspection task using Requicha's ideas can be done in the same way that was discussed in Section

4.6.3.

In the case of the conventional inspection the determination of the appropriate sensor can be done using simulations and Equation (4.4.67), illustrated by Figure 4.59. This figure shows the minimum size of tolerance zone that is possible to inspect in the case of the hole H of Figure 4.46 with $P_f^{feat} = P_n^{feat} = 0.01$, pixels of size *0.3 mm* × *0.28 mm* and for different values of σ_S.

As we cannot expect pixels sizes much smaller than *0.28 mm* × *0.3 mm* and the errors in the estimate of the radius do not vary much with the actual value of the radius, this figure gives a good idea of the limits in the conventional hole size inspection using the algorithm described in Section 4.3.3.

4.6.8 Hole pose inspection

We will use the expression hole pose inspection to designate the application of the I_p^{pose} diagnosis (see Section 4.5) to the curve created by the intersection between a manufactured planar surface and a manufactured circular cylindrical surface. As before, we will only be concerned with the cases in which the normal of the planar surface, the axis of the cylindrical surface and the viewing direction are parallel.

In order to exemplify the implementation of the hole pose inspection procedure let us consider the inspection of the hole in Figure 4.45 (hole H in Figure 4.46) as we did in Sections 4.6.3 and 4.6.7. Furthermore, let us also assume that the datum used to determine the designed position of the hole's border is the center of the hole itself.

Under these assumptions, a possible implementation of the I_p^{pose} is the verification of the position of the edge points extracted from the hole's border with relation to a test tolerance zone with center at the extracted datum feature (center of hole's border). This approximation is only justifiable in the cases in which the error in the extraction of the datum feature is much smaller than the error in the determination of the edge points belonging to the hole's border.

In the case of hole H, the uncertainty region in the extraction of the center of the hole's border can be approximated by a disk with radius *0.085 mm* for a probability $P = 0.01$ (according to the result of 100 simulations analysed as discussed in Section

4.3.3). As we discussed in Section 4.6.3, for the same probability P the size of the uncertainty region associated with the extraction of the edge points is *0.9 mm*. This means that the uncertainty region associated with the determination of the hole's center is approximately 10 times smaller than the uncertainty region associated with the extraction of the edge points.

Therefore, the implementation of the I_p^{pose} using a unique test tolerance zone is justifiable in the case of hole H. Despite this fact, in order to illustrate the use of Equation (4.5.78), we will implement the I_p^{pose} diagnosis by considering different test tolerance zones with centers in a neighbourhood of the extracted datum feature (hole's border center).

As in Section 4.6.3, the pose of the centers of the different test tolerance zones will be determined using Equations (4.6.87) and (4.6.88) with $\Delta_{Pose} = 0.05\ mm$ and $N_{pose} = 41$. This means that each cell (i,j) in the grid can be associated with a probability $\delta Prob^{(i,j)}$ (see Section 4.5) equal to:

$$\Delta Prob^{(i,j)} = f_{\delta Pose}{}^{(i,j)} \ \Delta_{Pose}{}^2 \tag{4.6.90}$$

where: $f_{\delta Pose}{}^{(i,j)}$ is the value of $f_{\delta Pose}$ at the center of the cell (i,j).

The use of Equation (4.5.78) is only possible if we have an approximation of the probability density function $f_{\delta Pose}$ associated with the errors in the determination of the designed pose of the hole. In the case of the hole H the exact determination of this function is complicated by the fact that the extraction of the center of the hole's border is biased. However, due to the small value of the bias, it is reasonable to approximate $f_{\delta Pose}$ by a normally distributed 2D random vector with mean zero.

By applying Equation (4.3.6) to (4.3.8) to the result of 200 simulation we found out that the variance Var of the normally distributed random vector can be approximated by Equation (4.6.91).

$$Var = \begin{bmatrix} +4.65e-5 & -4.99e-6 \\ -4.99e-6 & +6.94e-5 \end{bmatrix} \tag{4.6.91}$$

The application of the I_p^{pose} to the range image of Figure 4.45 was only possible after the elimination of the wrong range measurement caused by the spurious reflections of the laser - see Section 4.6.3. As in the case of the hole shape inspection, we used 180

edge points, $T_{test} = 1.8$ mm and $F_{out} = 5$ which corresponds to $P_f = 1.0e-3$ and $P_n = 3.7e-4$ (assuming $T_{out} = 7$).

As a result of the pose inspection procedure we obtained $P_{OK} = 0.9996$ which means that the hole's border was considered within tolerance for all the positions of the test tolerance zone considered. This result agrees with our previous discussion about considering a unique test tolerance zone and ignoring the errors in the determination of the datum position (center of hole's border).

It is interesting to observe that under the same conditions, the inspection only rejected the hole's border for all positions of the test tolerance zone when we had T_{test} not bigger than *1.7 mm.*

Similarly to the case of hole size inspection and for the same reasons, the implementation of the I_p^{pose} diagnosis implies in the use of much bigger tolerance zones than it would be possible if we considered the conventional pose inspection in which we just verified the position of the center of the hole's border. In the case of hole H, for instance, we would be able to inspect tolerance zones approximately ten times smaller using the same sensor and the same values of reliability and sensitivity.

However, it is important to observe that the I_p^{pose} diagnosis is a much better approximation of the common practice in the manufactures in which the inspection is carried out by introducing a gauge pin inside the manufactured hole.

The determination of the appropriate sensor for implementing a given hole pose inspection follows the same ideas we discussed in Section 4.6.3.

4.6.9 Segment pose inspection

In this section we will be concerned with the application of the I_p^{pose} diagnosis to curves resulting from the intersection of two manufactured planar surfaces when both surfaces are visible in a unique range image.

In order to illustrate the implementation of the I_p^{pose} let us consider the inspection of the segment in Figure 4.44 that corresponds to the segment AB in Figure 4.46. Furthermore, let us consider the datum reference frame shown in Figure 4.46 with

an axis parallel to the segment AB and the origin in the vertex A. This assumption means that the vertex A and the direction of segment AB are the datum features used to specify the designed pose of the segment AB.

We will implement the segment pose inspection in a way similar to the way that we implemented the I_p^{size} diagnosis in Section 4.6.6. As in the previous section we have two different possibilities:

1. to ignore the error in the extraction of the datum features (vertex A and direction of segment AB in this case) and check the position of the set of measured features in relation to a unique tolerance zone positioned according to the measured datum features, and

2. to take into account the error in the determination of the datum features and consider different tolerance zones positioned in a neighbourhood of the pose obtained using the extracted datum features.

In the case of the segment inspection, the pose *Pose* of the center of the test tolerance zones corresponds to the coordinates of one of the vertices of the segment being inspected and the two angles θ and ϕ defining the direction of the line parallel to the segment. We will determine the position of the datum features using the algorithm described in Section 4.3.4 for the determination of the vertex position and the algorithm described in Section 4.3.6 for the determination of the direction of the segment AB.

The direction of the segment AB during the inspection procedure is determined by calculating the intersection of the least square approximation of the manufactured planes $ABCD$ and $ABEGF$ in Figure 4.44 (see Figure 4.46). The least square approximations of the planes were determined using respectively 1734 pixels and 2789 pixels as described in Section 4.3.5. According to the result of 200 simulations the maximum error in the determination of the angles ϕ and θ should be smaller than *2.0e-4 rad* under the same conditions as the inspection.

The determination of the vertex A during the inspection is done by the application of the algorithm of Section 4.3.4 (as it is described in Section 4.6.6). Using the result of 200 simulations carried out under the same conditions as the inspection and the

procedure described in Section 4.3.4, we modelled the errors in the determination of the vertex A as a 3D normally distributed random vector with mean zero and variance Var given by:

$$Var = \begin{bmatrix} +0.0500 & -0.0132 & -0.0080 \\ -0.0132 & +0.2000 & +0.1120 \\ -0.0080 & +0.1120 & +0.0800 \end{bmatrix} \quad (4.6.92)$$

The results of the simulation show that the errors in the determination of the angles ϕ and θ are relatively much smaller than the errors in the determination of the edge points of segment AB (see Section 4.6.1) and the errors in the determination of vertex A (an angle of *2.0e-4 rad* corresponds to an arc of size equal to only *1.8e-2 mm* if we assume a radius of *90 mm* that is the nominal size of the segment AB).

Therefore, we decided to consider just the errors in the determination of the vertex A during the implementation of the I_p^{pose}. This means that the inspection procedure will consider different test tolerance zones defined in such a way that:

- the center of the test tolerance zones will be segments whose direction will be defined by the measured values of the angles ϕ and θ and
- the center of the test tolerance zones will start from vertices (x_p^i, y_p^j, z_p^k) belonging to a grid defined in a similar way to Equations (4.6.80) to (4.6.85), but centered in the measured value of the vertex A.

As a consequence of this fact, the I_p^{pose} diagnosis will be implemented taking into account only the probability density function of the errors in the extraction of the vertex A ($f_{\delta Pose}$). This means that we will consider $\delta Prob^{(i,j,k)}$ equal to:

$$\Delta Prob^{(i,j,k)} = {f_{\delta Pose}}^{(i,j,k)} \ {\Delta_{pose}}^3 \quad (4.6.93)$$

where: ${f_{\delta Pose}}^{(i,j,k)}$ is the value of $f_{\delta Pose}$ at the center of the cell (i, j, k).

Using the results of the simulation discussed previously, the probability density function will be approximated assuming the errors normally distributed with mean zero and variance given by Equation (4.6.92).

The results of the application of the pose inspection procedure to the range image of Figure 4.44 (under the same conditions of the size inspection of Section 4.6.6, but with

T_z (mm)	I_p^{pose}	K_c MAX
3.6	0.001	∞
3.8	0.33	10.0
4.0	0.48	5.0
4.2	0.66	3.3
4.4	0.80	2.5
4.6	0.89	2.0
4.8	0.95	1.3

Table 4.11: Value of T_z and corresponding values of I_p^{pose} and maximum value of process capability index K_c in the inspection of segment AB of Figure 4.46.

$N_{pose} = 41$) can be summarised by Table 4.11. This table shows the value of I_p^{pose} obtained for different values of T_z, as well as the maximum value of process capability index K_c associated with each value of T_z. The maximum value of K_c is the maximum value of the process capability index for which the value of T_z (according to Equation (4.4.67)) is not smaller than the corresponding value of T_z shown in the table.

As we can see from the table, according to the result of the inspection the segment AB can only be considered within tolerance with reasonable values of certainty (I_p^{pose} bigger than 0.9) for values of the process capability index smaller than 1.3.

It is important to observe that the approximation of the value of P_{OK} shown in Table (4.11) is not exact. This happens mainly due to the errors in the determination of $f_{\delta Pose}$ and the errors in caused by the discretisation of Equation (4.5.77).

4.7 Conclusions

In this chapter we presented a method for the designing and implementation of inspection procedures that allow not only the verification of the most common tolerances defined in the standards, but also the verification of tolerances defined according to Requicha's theory of tolerancing. The method proposed the implementation of inspection procedures using three different steps: feature extraction, tolerance zone positioning and inspection diagnosis.

A theoretical model of the reliability and sensitivity of the inspection procedures based

on the statistical modelling of the errors in the feature extraction phase using Monte Carlo simulations was presented. A formal definition of the inspection diagnosis was also presented.

The application of the method in the inspection of some common simple geometric elements (straight lines, planes and circles) was discussed in details from the design of the feature extraction algorithms to the implementation of the inspection procedures. Examples with real and synthetically created images were shown to illustrate the method.

Chapter 5

Registration

This chapter is concerned with the registration between sculptured surfaces and its main objective is to analyse the use of the ICP algorithm for inspection purposes (see Section 2.3.4 for a description of the ICP algorithm). A preliminary version of this chapter can be seen in [Bispo & Fisher 94b] and [Bispo & Fisher 94a].

Section 5.1 introduces the chapter by discussing the objectives of the registration process and the importance of its accuracy in different inspection tasks. Section 5.2 presents the model shapes we used in our experiments. The limits of the accuracy of the ICP algorithm are discussed in Section 5.3. Section 5.4 presents the implementation of a modified version of the ICP algorithm for inspection purposes. Section 5.5 presents the results of our experiments with the ICP algorithm. Our final comments are presented in Section 5.6.

5.1 Introduction

After data acquisition, the next step in the inspection of a manufactured object is the alignment or registration between the image of the object being inspected and the model of the nominal shape of the object. This process consists in determining the rigid transformation (rotation $\mathbf{R}$ and translation $\mathbf{T}$) that superposes model and data.

The inspection of machined objects might involve the registration between model and data in two different situations:

1. The object being inspected has a number of distinctive features (*e.g.* planes), enough to make it possible to obtain the registration between model and data. In this case the registration can be obtained using the matching techniques (*e.g.* [Faugeras & Herbert 86] and [Grimson & Lozano-Perez 84]) that we discussed in Section 2.3. This case corresponds to the majority of the common manufactured objects and it was discussed in detail in [Marshall 89] - see Section 2.1. A discussion of registration in this situation is outside the scope of this thesis.

2. In the case of sculptured surfaces such as turbine blades, aerodynamic surfaces or cameras, we may not have enough salient features that could help the registration process and we must concentrate solely on the use of 3D points for obtaining the registration. The main objective of this chapter is to investigate the use of the ICP algorithm (see Section 2.3.4) for obtaining the registration between model and data in this case.

An important difference between the two cases is the relative importance of the accuracy of the registration process for the inspection. In the first case, we are generally concerned with the inspection of some particular features of the object being inspected. This means that we are concerned with the shape, size and the position of these particular features in relation to other features of the range image.

Therefore, as we discussed in Chapter 4, the accuracy of the registration process is not critical. The registration must only produce an estimate of the position of the features in the range image that is good enough to guide the feature extraction process during the inspection procedures.

For instance, let us consider the inspection of the straightness of a segment corresponding to the intersection of two manufactured planar surfaces. The objective of the registration is to determine an initial estimate of the position of the edge points in the range image. The final reliability, sensitivity and accuracy of the straightness inspection depends only on the accuracy and precision of the determination of the 3D points belonging to the segment being inspected.

It is important to observe, however, that inspection does benefit from increases in the accuracy of the registration process because, in general, it causes improvements in the

performance of the feature extraction process.

In the inspection of sculptured surfaces the main objective is the evaluation of the shape of the surface and we are not concerned about any particular features. This means that the inspection of sculptured surfaces can be carried out by looking at the difference between the 3D points of the model and the corresponding 3D points of the data after the registration. Thus, the accuracy of the registration process in this case is critical and the errors during the registration phase might lead to a wrong inspection diagnosis.

In the next sections we discuss the registration of sculptured surfaces using the ICP algorithm in detail. We start the chapter by describing the model shapes of sculptured surfaces that we used in our experiments.

5.2 Model shapes

We decided to use B-splines for modelling the nominal shape of sculptured surfaces. This decision was based on the fact that B-splines are commonly used for modelling the shape of sculptured surfaces.

In all the experiments with sculptured surfaces described in this thesis, we modelled the nominal shape of our test objects (see Figures 5.1 and 5.2) using non-rational uniform B-splines. The models were extracted from range images of the test objects using the following procedure:

- **Smoothing**: Application of conservative and average smoothing to the range image in order to eliminate outliers.
- **Thresholding**: Thresholding of the original image for separating background from foreground and delimiting holes. Holes were delimited by thresholding the range image only in a window containing the hole. The thresholding was used for producing a bitmap indicating the interesting points in the range image.
- **Fitting**: Fitting of a tensor uniform B-spline to the interesting points indicated by the bitmap produced by the thresholding. The fitting was done by using an

algorithm proposed in [Deboor 78] that minimises the average square distance between the spline model and the image points.

Because the direct use of the B-spline models in the implementation of the ICP algorithm would involve big computational costs that would make the algorithm very slow, we implemented the ICP algorithm using sets of 3D model points obtained by sampling the B-spline models. More details related to this procedure are given in the next sections.

In all our experiments we used two test objects that we will call Object **1** and Object **2** which are shown in the range images in Figures 5.1 and 5.2.

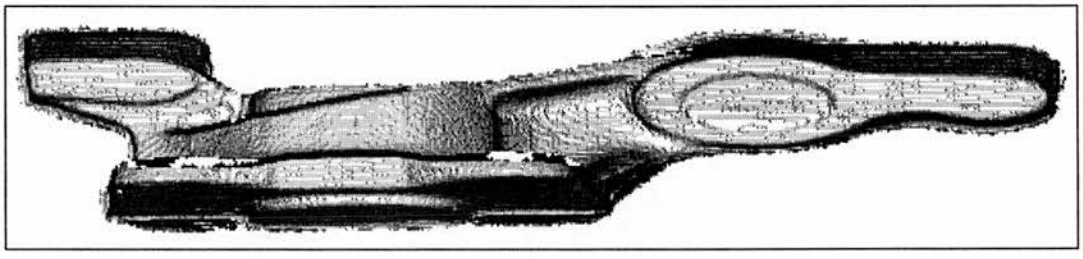


Figure 5.1: Range image of Test object number 1.

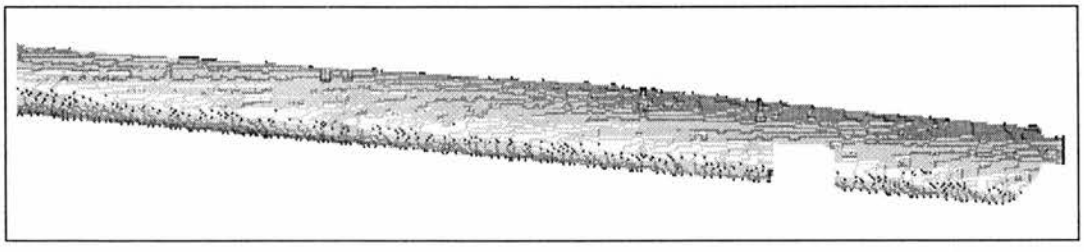


Figure 5.2: Range image of Test object number 2.

Figures 5.3 and 5.4 show the spline models built for the test objects **1** and **2**. The average distance between model points and data points in both cases were on the order of tenths of millimetres (with the image noise standard deviation equal to *0.15 mm*). As one would expect, the biggest modelling errors happen near the discontinuities of the range image. This fact explains the better results obtained with test object number **2** which has less discontinuities than test object number **1**.

From now on, when we use the spline models shown in Figure 5.3 and 5.4 we will use the average distances between model and data (respectively equal to *0.08 mm* and *0.4*

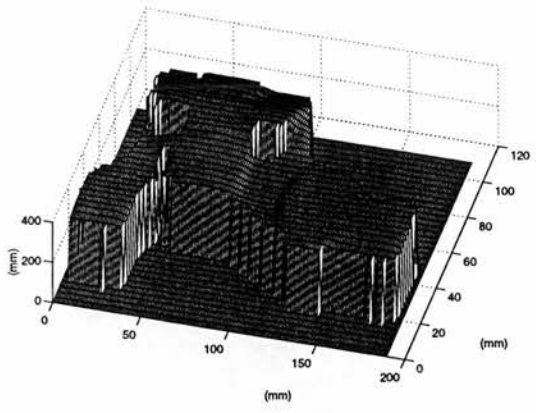


Figure 5.3: Uniform B-spline model of Test object number 1. Average distance between model and data equal to *0.4 mm*. Uniform grid of *50* × *50* knots.

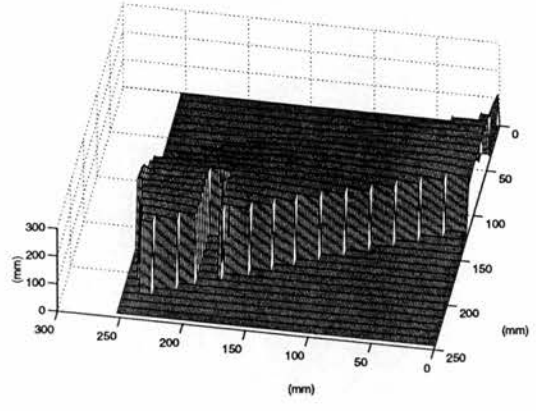


Figure 5.4: Uniform B-spline model of Test object number 2. Average distance between model and data equal to *0.08 mm*. Uniform grid of *50* × *50* knots.

mm for the two objects) as the standard deviations of the random processes describing the combined effect of the sensor and modelling errors associated with Figures 5.3 and 5.4.

This procedure is justified by the fact that the values of average distance correspond to the standard deviation of the difference between the points of the range images and their respective B-spline models.

5.3 Maximum accuracy in the registration of sculptured surfaces

The maximum accuracy we can achieve with the ICP algorithm is limited by the accuracy of the operator **Q(P,Y)** described in Section 2.3.4. The maximum accuracy

corresponds to the case in which all the correspondences between model points and data points are correct and we do not have any sampling errors.

Under these conditions, the error in the determination of the rotation $\mathbf{R}$ and translation $\mathbf{T}$ aligning the model points and data points depends only on the robustness of the operator $\mathbf{Q(P,Y)}$ to the errors in the range measurement.

In order to investigate the accuracy of the operator $\mathbf{Q(P,Y)}$, let us consider a set of data points $\mathbf{P} = \{\mathbf{p_i}\}_{\mathbf{i=1,n}}$ and a set of corresponding model points $\mathbf{Y} = \{\mathbf{x_i}\}_{\mathbf{i=1,n}}$ such that $\mathbf{P}$ and $\mathbf{Y}$ are related by a rigid transformation - as defined in Equations (5.3.1) to (5.3.3).

$$\mathbf{p_i} = \mathbf{R}\mathbf{x_i} + \mathbf{T} + \eta_i \tag{5.3.1}$$

$$\mathbf{E}[\eta_i] = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{5.3.2}$$

$$\mathbf{E}[\eta_i \eta_i^T] = \begin{bmatrix} \delta_i^2 & 0 & 0 \\ 0 & \delta_i^2 & 0 \\ 0 & 0 & \delta_i^2 \end{bmatrix} \tag{5.3.3}$$

where:

- η_i is a random vector process normally distributed that models the sensor and modelling errors,
- η_i^T is the transpose of η_i,
- $\mathbf{E}[\cdot]$ is the expectation operator and
- δ_i is the standard deviation associated with the η_i

The problem of finding $\mathbf{T}$ and $\mathbf{R}$ that minimise the average square distance E_M between $\mathbf{P}$ and $\mathbf{Y}$ (see Equation (5.3.4)) is a well known problem to which there are several proposed solutions such as [Faugeras & Herbert 86], [Horn & Harrys 87], [Arun *et al.* 87] and [Walker *et al.* 91]. Under the criterion of accuracy in finding the right values of $\mathbf{R}$ and $\mathbf{T}$ all these methods are equivalent [Lorusso & Eggert 95], and we will concentrate our attention on [Horn & Harrys 87].

The estimate of $\mathbf{T}$ in Horn's algorithm is calculated using the estimate of rotation $\hat{\mathbf{R}}$ and the centers of mass of the sets of model points and data points. This means that

the errors of the estimate $\hat{\mathbf{T}}$ are the result of the propagation of errors in all the stages of the algorithm. Because of this fact and the intuitive meaning of the errors in the estimate of translation, we will use the value of $\|\mathbf{T} - \hat{\mathbf{T}}\|$ to evaluate the accuracy of the registration.

$$E_M = {D_M}^2 = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{p_i} - \mathbf{R}\mathbf{x_i} - \mathbf{T}\|^2 \tag{5.3.4}$$

To investigate the robustness of the operator $\mathbf{Q(P,Y)}$ we used Monte Carlo simulations in which we synthetically created the sets $\mathbf{P}$ and $\mathbf{Y}$ of corresponding points in such a way that these sets were related by a known rigid transformation (rotation $\mathbf{R}$ and translation $\mathbf{T}$). Then, we observed the values of E_M and the errors in the estimate of $\mathbf{R}$ and $\mathbf{T}$ as a function of δ_i and the number of matched points.

The model points $\mathbf{Y}$ were extracted from the spline models of the test objects **1** and **2**. The data points $\mathbf{P}$ were generated by applying a known rotation and translation to the model points and corrupting the result with a normally distributed random vector process with mean zero and variance defined by Equation (5.3.3).

The experiment was run for different values of rotation and translation and different values of standard deviation δ_i. Two hundred simulations were executed for each value of δ_i.

As one could expect the results did not depend on the specific values of rotation and translation. Also, the final value of the mean square distance E_M found by Horn's algorithm tended asymptotically to $\mathbf{E}[E_M] = 3{\delta_i}^2$ as the number of matched points increased.

Figures 5.5 and 5.6 show the variation of D_M, see Equation (5.3.4), and the error of the translation estimate ($\hat{\mathbf{T}}$) for the test object number **1**. Figures 5.7 and 5.8 show the same for the test object number **2**.

These graphs show the maximum accuracy that can be achieved for a given level of noise, which corresponds to the combined effects of sensor and modelling errors, and number of matched points. A careful comparison of the two sets of graphs shows that the accuracy of the translation estimates does not change much with the shape of the surface being registered. The graphs also show that the error in the estimate of

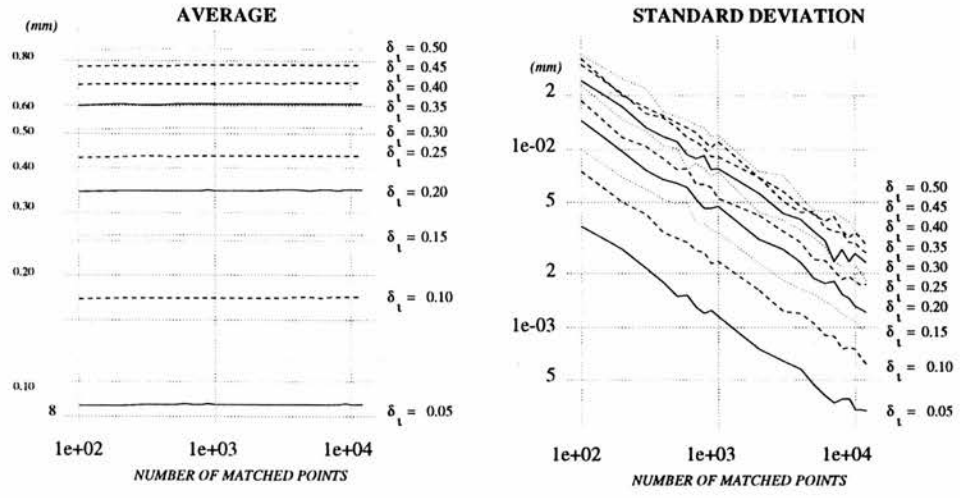


Figure 5.5: Variation of the mean distance D_M with the noise level δ_i for test object number **1**. Average and standard deviations calculated using 200 samples (10 different curves for δ_i varying from *0.05 mm* to *0.5 mm*).

translation will decrease almost inversely with the number of matched points, as shown in Equation (5.3.5).

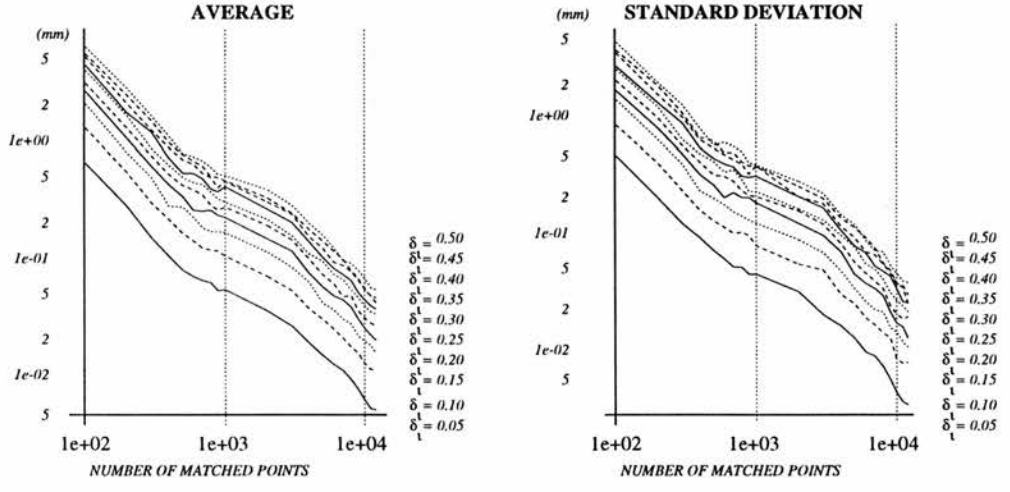


Figure 5.6: Variation of the translation error $\|\mathbf{T} - \hat{\mathbf{T}}\|$ with the noise level δ_i for test object number **1**. Average and standard deviations calculated using 200 samples (10 different curves for δ_i varying from *0.05 mm* to *0.5 mm*.

It is important to remember that the graphs show the very best accuracy that can be achieved, because other sources of errors such as sampling errors, modelling errors and mismatchings were not taken into account.

$$\|\hat{\mathbf{T}} - \mathbf{T}\| \quad \propto \quad \delta_i / N \tag{5.3.5}$$

The results shown in Figures 5.5 to 5.8 are valid only in the case in which the variance

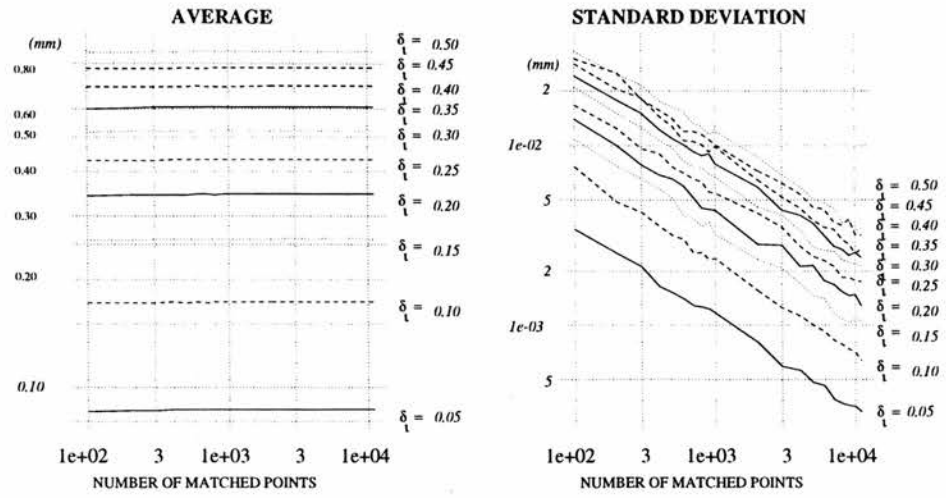


Figure 5.7: As Figure 5.5 but for test object number **2**.

of the random process η_i can be described by Equation (5.3.3). This means that the results of the simulation can only be used when the sensor errors are isotropic, *i.e.* there is not a direction in which the sensor errors are expected to be bigger or smaller.

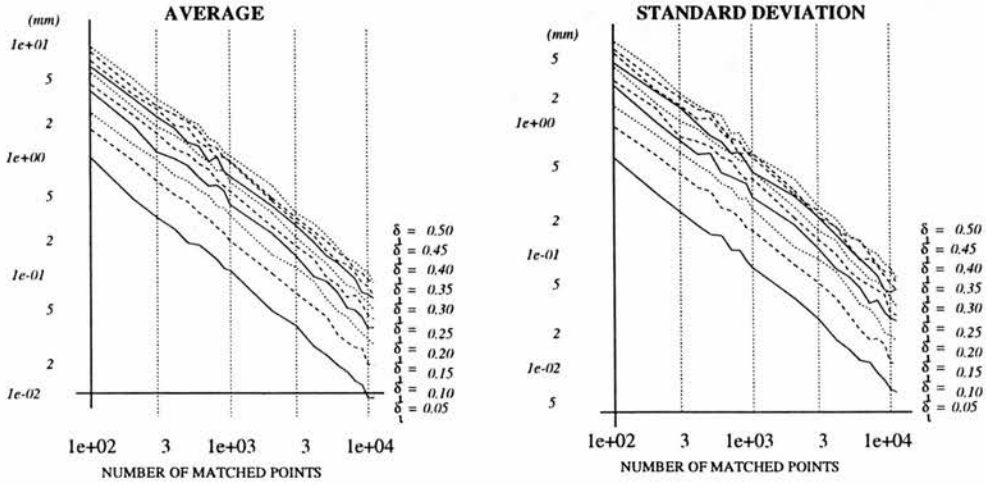


Figure 5.8: As Figure 5.6 but for test object number **2**.

In the case of laser stripers, the model of sensor errors described by Equation (5.3.3) is not valid and a more realistic model should consider that we have errors just in the range measurement (z coordinate in sensor reference frame). When the sensor errors are anisotropic, the performance of Horn's algorithm is basically the same, but with a degradation of approximately *15* % in the accuracy.

This means that if we have two different sensors such that:

1. The errors of the sensors can be modelled by two random vector processes η_i and η_i' with the same energy, *i.e.*:

$$\mathbf{E}[{\eta_i}^T \eta_i] \;=\; \mathbf{E}[{\eta_i'}^T \eta_i'] \tag{5.3.6}$$

2. η_i is isotropic and η_i' is non-zero only in the z direction.

Then, the errors in the determination of $\hat{\mathbf{R}}$ and $\hat{\mathbf{T}}$ will be approximately *15* % bigger in the case of η_i'.

The degradation in the performance of the algorithm in the case of errors that are not isotropic can be explained by the fact that the deduction of Horn's algorithm (the same is valid for the deduction of the algorithms described in [Faugeras & Herbert 86], [Walker *et al.* 91] and [Arun *et al.* 87]) is based on the minimisation of the average distance between model points and data points after alignment.

Therefore, the rotation $\hat{\mathbf{R}}$ and translation $\hat{\mathbf{T}}$ are determined in such a way that the differences in the coordinates of the model points and the data points in different coordinates (directions) have the same relative weight, *i.e.* the errors are implicitly assumed to be isotropic.

Figures 5.5 to 5.8 illustrate how it is possible to use Monte Carlo simulations to determine the maximum accuracy we can achieve with the ICP algorithm applied to a given model shape $\mathbf{X}$ as a function of the sensor and modelling errors (δ_i) and the number of sampling points used.

As the modelling errors are in most of the practical situations null or negligible, the Monte Carlo simulations can be used to determine the maximum accuracy of the algorithm as a function of the errors in the range measurement (sensor) and the number of sampling points.

The degradation in the accuracy of the ICP algorithm caused by correspondence and sampling errors is difficult to model as it depends on a considerable number of factors such as: the model and data shape, the number of wrong matches with relation to the total number of correct matches, the size of the errors caused by the sampling, etc.

In the next section we present our implementation of the ICP algorithm. We propose

the use of consistency checks to eliminate wrong matches and, consequently, to reduce the degradation caused by the correspondence errors. Furthermore, we also propose the use of a subpixel technique that aims to reduce the effects of the sampling errors and improve the final accuracy of the algorithm.

5.4 Implementing the ICP

In our implementation of the ICP algorithm, which we will call modified ICP, we followed basically the procedure described in Section 2.3.4 and we considered two different sets of 3D points: a set of data points $\mathbf{P}$ corresponding to the range image and a set of model points $\mathbf{Y}$ obtained by sampling of the B-spline models.

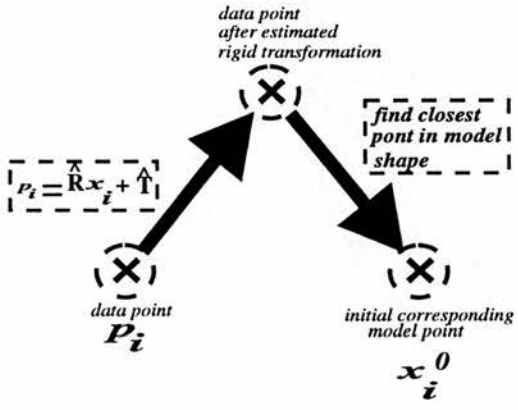


Figure 5.9: Using the estimates of rotation ($\hat{\mathbf{R}}$) and translation ($\hat{\mathbf{T}}$) to determine the initial estimate of the corresponding point.

Taking into account the variation in the maximum accuracy of the algorithm with the number of matched points considered and the excessive computational costs in the implementation of the ICP with too many pairs of corresponding points, we adopted a multiscale approach in which we considered two distinctive phases:

1. **Coarse approximation**: In this phase the objective is to obtain a first approximation of the rotation $\mathbf{R}$, the translation $\mathbf{T}$ and the correspondences between the shape model and the data points.

   The set of model points $\mathbf{Y}$ is determined using a coarse sampling grid of the model shape $\mathbf{X}$ and the range image is subsampled. The model shape $\mathbf{X}$ was sampled using a regular rectangular grid of sampling points in the x and y directions. For

objects of the same dimensions as the test objects (approximately equal to *20 cm* × *10 cm* × *10 cm*) typical values are a model sampling grid of size *1 mm* × *1 mm* and a data sampling grid of size approximately two to four times bigger.

This means that we use approximately *1000* matched points which, see Figures 5.6 and 5.8, corresponds to an accuracy of the order of a tenth of a millimetre.

2. **Fine approximation:**

In this phase the number of data points is increased and smaller sizes of model sampling grids are used. The whole process can be described by:

(a) **Increasing of number of data points:**

A finer grid of data points is created and each data point is associated with an initial estimate of its corresponding model point. The initial estimate of the corresponding model point is determined by applying the current estimates of $\mathbf{R}$ and $\mathbf{T}$ to each data point and associating each data point to the closest model point in the original coarse grid of model points.

Figure 5.9 illustrates this process. Notice that the maximum number of data points is limited by the number of model points in the coarse sampling of the model shape.

(b) **Subpixel technique for improving accuracy:**

New estimates of the model points corresponding to each data point are obtained by considering finer sampling grids of the model shape. In this phase the ICP algorithm is followed normally for a few iterations during which the size of the model sampling grid is reduced to half of its size in the previous iteration.

The search for the closest model point $\mathbf{x_i}^k$ corresponding to the data point $\mathbf{p_i}$ at iteration k is done only in a neighbourhood of $\mathbf{x_i}^{k-1}$ (estimate of the closest point to p_i at iteration $k-1$) - see Figure 5.10.

The new closest point $\mathbf{x_i}^k$ is determined by search in a grid centered on $\mathbf{x_i}^{k-1}$ with a typical total of 25 points.

A fundamental stage in the implementation of the ICP algorithm is the definition of

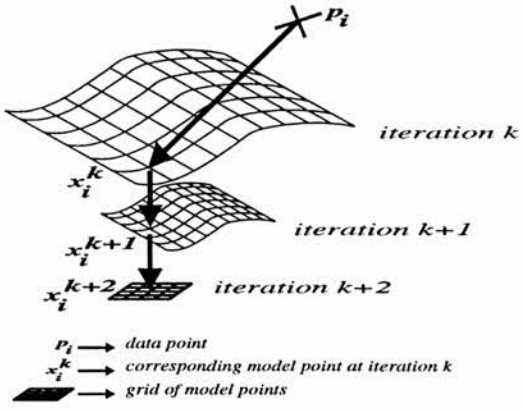


Figure 5.10: Improving the estimates of rotation ($\hat{\mathbf{R}}$) and translation ($\hat{\mathbf{T}}$) using finer sampling grids.

the operator **C(P,X)** (see Section 2.3.4) that finds the set of model points **Y** which are the closest points in the model shape **X** to the data points **P**.

Many different approaches were adopted in the implementation of this operator by different researchers. Zhang [Zhang 93], for instance, used K-D trees in the implementation of a registration algorithm similar to the ICP. A good discussion of the problem can be seen in [Bentley *et al.* 80] in which the use of Voronoi diagrams is proposed.

In our case, during the first phase of the registration (coarse approximation) we opted for a very simple implementation of the **C(P,X)** operator. The coarse grid of model points is searched in a multiscale fashion and the search for the closest point is done in two different stages.

In the first stage an approximation of the closest point is found by exhaustive search in a subsample of the grid of model points. In the second stage the final approximation is found by exhaustive search in a neighbourhood of the initial approximation of the closest point.

This approach can occasionally fail when the region containing the real closest point is not correctly determined during the first stage of the search. In our experiments this never happened because we always started the search at a scale small enough to avoid this problem. Also, due to the consistency checks we added to the ICP algorithm (see Section 5.4.1), occasional failures to find the true closest point do not degrade the

performance of the algorithm.

During the second phase of the registration (fine approximation) the closest point is determined by exhaustive search in a small grid of model points centered on the current estimate of the closest point.

An important limitation in the ICP algorithm is the fact that the algorithm cannot cope with situations in which the data points are not a subset of the model points. This fact represents a problem for the use of the ICP in the inspection of sculptured surfaces because the eventual defects (deformations) on the data surface will cause a degradation in the accuracy of the registration.

In order to alleviate this problem and to make the algorithm more robust to outliers, we altered the original algorithm by eliminating some of the pairs of corresponding data points and model points found by the $\mathbf{C}(\mathbf{P_k}, \mathbf{X})$ operator at each iteration k (see description of ICP algorithm in Section 2.3.4).

The idea is to eliminate the correspondences found by the $\mathbf{C}(\mathbf{P_k}, \mathbf{X})$ operator that do not satisfy some consistency checks and, therefore, are clearly wrong. We used two different kinds of consistency checks:

1. **Based on the current estimate of rigid transformation**: This first kind of consistency check (also used in [Zhang 93] though in a different way) eliminates pairs of corresponding data points and model points if the distance between these points is bigger than three times the value of D_M found so far in the registration - see Equation (5.3.4).

2. **Based on local geometric constraints**: These tests use the distance between points in the sampling grid and the angles between the vectors joining pairs of points in the sampling grid to eliminate wrong correspondences.

The consistency checks are used during the first phase of the registration (coarse approximation) and they are only applied after the value of D_M found so far in the registration is smaller than at most ten times the expected value of D_M after the perfect alignment between model and data.

In the cases of the test objects **1** and **2**, for instance, the consistency checks are only applied after the value of D_M found in the registration process is smaller than *3.0 mm* and *1.0 mm*, respectively.

Thus, the consistency checks are used only in the final iterations of the first phase of the registration when the correspondences between model and data points are expected to be approximately correct. More details on the implementation of the local geometrical constraints are given in the next section.

5.4.1 Local geometric consistency checks

Because of the surface sampling, a correct matching for a data point $\mathbf{p_i}$ is any model point $\mathbf{x_i}$ such that, after the alignment between model and data, $\mathbf{x_i}$ belongs to the portion of the data surface that $\mathbf{p_i}$ represents.

We approximated this region by a cuboid centered at $\mathbf{p_i}$ with the faces parallel to the planes **xy**, **xz** and **yz** of the sensor coordinate system and dimensions corresponding to the size of the sampling grid - see Figure 5.11.

The dimensions of the cuboid C_i associated with the data point $\mathbf{p_i}$ on the x and y directions are respectively equal to the size of the data sampling grid on these directions: Δ_x and Δ_y - see Figure 5.11.

The size of the cuboid C_i in the z direction, which we will call Δ_i^z, can be approximated using the values of f_x and f_y that are the first derivatives of the object surface in the x and y directions respectively - see Equation (5.4.7) and Figure 5.11.

$$\Delta_i^z = |f_x(x,y)|\Delta_x + |f_y(x,y)|\Delta_y \tag{5.4.7}$$

In the implementation of the consistency checks we determined the values of f_x and f_y using the normal of the data surface at the point $\mathbf{p_i}$. The normal was estimated by approximating a small neighbourhood of the data points surrounding the data point $\mathbf{p_i}$ by a plane - see Equations (4.3.45) to (4.3.47).

Neighbourhoods with between *16* to *25* data points were used. Furthermore, we also used Δ_i^z *10* % to *20* % bigger than the value given by Equation 5.4.7 to compensate for the sampling and sensor errors.

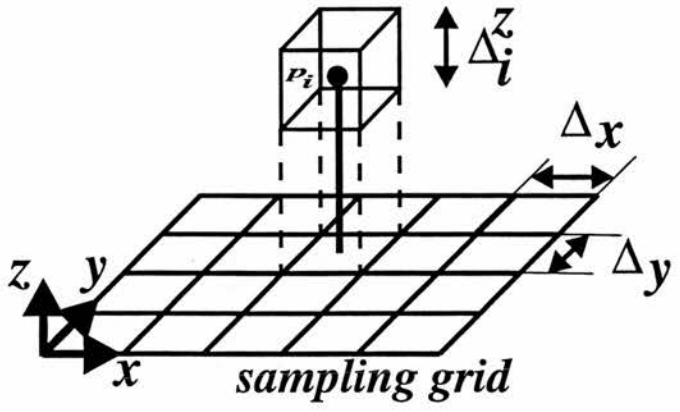


Figure 5.11: Dimensions of cuboid C_i associated with data point $\mathbf{p_i}$.

In order to understand the idea behind the consistency checks, let us consider two data points $\mathbf{p_i}$ and $\mathbf{p_j}$ and their respective cuboids C_i and C_j. The distance between two model points corresponding to these two data points must not be bigger than the maximum distance between two 3D points defined in such way that the first belongs to C_i and the second to C_j. The distance between the two corresponding model points must also not be smaller than the minimum distance between the two cuboids.

Figure 5.12 illustrates this idea in the two dimensional case when the cuboids correspond to rectangles.

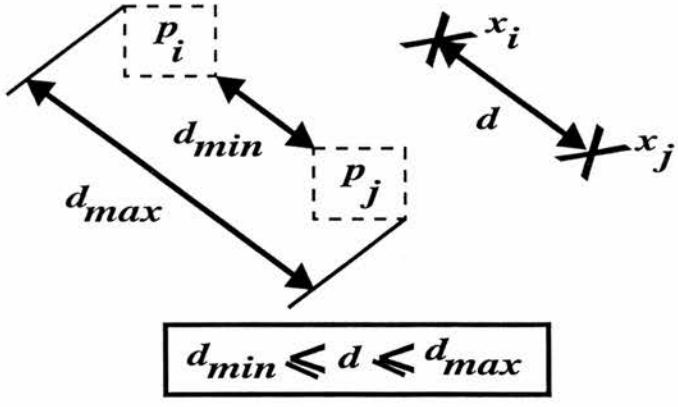


Figure 5.12: Illustration of the distance consistency check among the data points $\mathbf{p_i}$ and $\mathbf{p_j}$ and the corresponding model points $\mathbf{x_i}$ and $\mathbf{x_j}$ in the two dimensional case.

If we consider three data points, it is also possible to use the cuboids associated with these data points to determine the maximum and minimum angle between vectors joining the corresponding model points - see Figure 5.13 for an illustration in the 2D

case.

We can conclude that the implementation of the consistency checks implies in a pre-processing stage in which we:

- determine the cuboids associated with each data point,
- calculate the maximum and minimum distance between all the pairs of cuboids and
- calculate the maximum and minimum angles between the vectors joining all the triplicates of cuboids.

During the registration all the pairs and triples of corresponding points found by the $\mathbf{C}(\mathbf{P_k}, \mathbf{X})$ operator are checked to verify that the distances and angles between the model points are within the limits of the values calculated during the pre-processing stage.

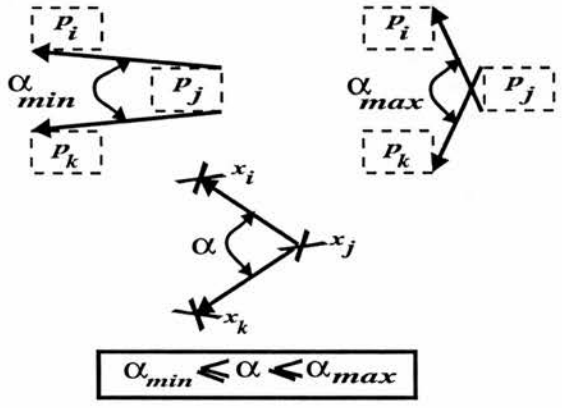


Figure 5.13: Illustration of the angle consistency check among the data points $\mathbf{p_i}$, $\mathbf{p_j}$ and $\mathbf{p_k}$ and the corresponding model points $\mathbf{x_i}$, $\mathbf{x_j}$, and $\mathbf{x_k}$ in the two dimensional case.

5.5 Experiments with the modified ICP

In our experiments with the modified ICP we observed all the properties described in [Besl & McKay 92] for the case in which the set of data points correspond to a subset of the set of the model points.

We started our experiments with simulations to determine the maximum accuracy in the registration estimate as a function of the standard deviation of the errors in the range measurement σ_S. The Monte Carlo simulations were run using 2000 pairs of model points and corresponding data points. The 3D points were created as described in Section 5.3 using a non-isotropic model of sensor errors.

Figure 5.14 and 5.15 show the result of the Monte Carlo simulations in the case of the test object **1**. The graphs show the average values of D_M and $\|T - \hat{T}\|$ over two hundred simulations. The corresponding standard deviation of the results are at least ten times smaller than the values shown in the graph.

The graphs illustrate the order of magnitude of the final accuracy that can be achieved considering a model sampling grid of *0.3 mm* × *1 mm* when the modified ICP converges.

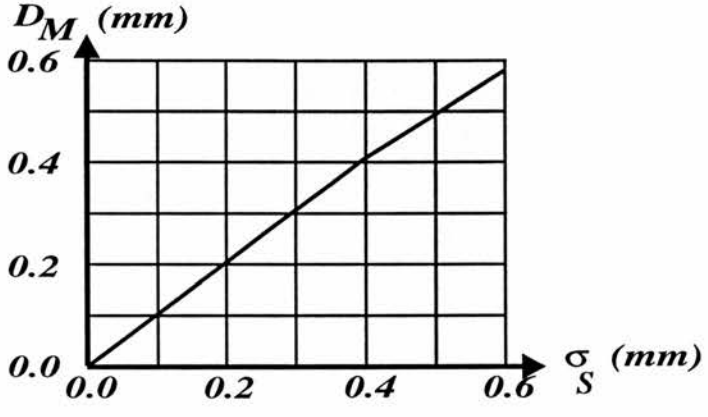


Figure 5.14: Variation of value of D_M with standard deviation σ_S. Average value resulting from 200 simulations using 2000 data points and model sampling grid of size *0.3 mm* × *1 mm*. Results valid for an initial pose between model and data for which the ICP converges to the right answer.

The graph in Figure 5.14 shows that the average value of D_M was approximately equal to the value of σ_S which indicates that the modified ICP really converged to the right answer for all the different values of σ_S and in all runs of the simulation.

The comparison between the errors in the estimate of the translation $\|T - \hat{T}\|$ in Figures 5.15 and 5.6 reveals that the error in the estimate of the translation is near to the minimum possible error of Horn's algorithm under the same circumstances (approximately 10 % to 15 % bigger).

This means that we must not have many mismatching errors in the final solution

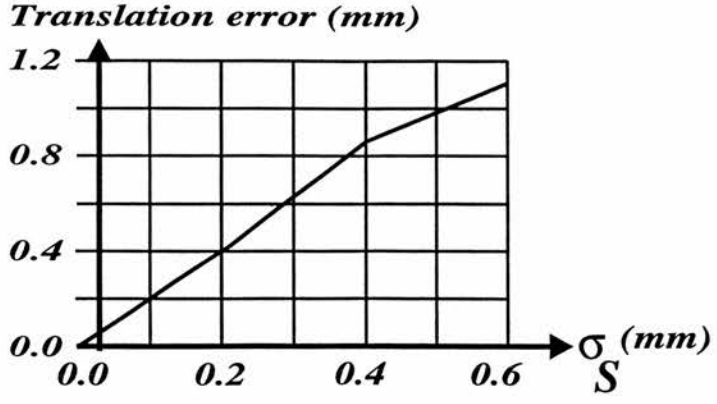


Figure 5.15: Error of estimate of translation $\|T - \hat{T}\|$ under the same conditions of Figure 5.14.

found by the ICP. Furthermore, the result also illustrates an interesting property of the algorithm that we observed in our simulations: the use of a model sampling grid with a size not bigger than *1 mm* and a data sampling grid not smaller than the model sampling grid eliminates most of the effects of sampling errors.

The results of similar simulations with the test object number **2** (in which the modified ICP converged to the right answer) are basically the same with a final accuracy of the same order of magnitude in the registration estimate.

The performance of the modified ICP with the two test objects, however, was not exactly the same. In the case of test object number **1** the biggest error in the estimate of translation occurred always in the x direction. In the case of test object **2** the biggest error in the translation estimate occurred always in the y direction.

This difference can be explained by the difference in shape between the two objects. A fast look at Figure 5.1 (test object **1**) is enough to suggest that we have the biggest variations in the values of range and in the normal direction in the y direction. This means that the algorithm will be more sensitive to errors in the determination of the component of the translation in the y direction.

In the case of test object **2** (see Figure 5.2) the object is almost flat and much bigger in the x direction than in the y direction. These facts make the algorithm more sensitive to errors in the determination of the x component of the translation.

These results illustrate how the distribution of the errors in the estimate of pose of

the ICP algorithm depends fundamentally on the shape of the object being registered. The errors will always be bigger in the direction in which the algorithm is less sensitive to the errors in the registration estimate.

A very important limitation of the ICP [Besl & McKay 92] algorithm is the fact that the algorithm might not converge to the right answer depending on the model shape, the size of the data shape and the relative pose between the model points and the data points at the first iteration of the ICP (which, in our case, corresponds to the initial estimate of pose used). This is a very important limitation of the algorithm that we also observed in our experiments with the modified ICP.

The ICP algorithm might converge to the wrong answer in two distinctive situations (see [Besl & McKay 92] and [Sahoo & Menq 91]):

- **Local minimum**: The algorithm stops in a local minimum of the value D_M that is not the global minimum of D_M and, therefore, does not correspond to the correct value of registration. This is the most common cause of error with dense range data.
- **Singularity**: The shape of the surface is such that there are different rigid transformations for which the value of D_M is minimum. In this case minimum value of D_M does not necessarily correspond to the right registration.

In order to assess the magnitude of the problems with the convergence of the ICP, we proceeded with experiments in which we used the ICP to register the range images of the test objects and their corresponding B-spline models (see Figures 5.1 to 5.4). The idea was to build a mapping function relating the initial relative poses between model and data and the error of the estimate of registration of the ICP algorithm (when applied to the test objects).

An interesting aspect of the influence of the initial estimate of pose in the performance of the ICP is the influence of the original estimate of translation. According to [Besl & McKay 92], when the data set is a subset of the model that covers a reasonable portion of the model, the ICP performance is reasonably insensitive to the original estimate of translation.

Although this observation is essentially true, in our experiments we observed that the final accuracy of the registration found by the algorithm can vary considerably as is illustrated in Figure 5.16 (A). This figure shows the value of the error in the estimate of the translation $\mathbf{T}$ in several runs of the ICP that differ only in the initial estimate of the value of $\mathbf{T}$ used. The graph shows that the error in the estimate of the translation can vary from *0.2 mm* to *1.0 mm*.

The situation becomes even worse when the data covers only a small part of the model as we can see in Figure 5.16 (B). In this case the data points cover only 50 % of the model surface and the errors in the estimate of the translation vary from *1 mm* to *12 mm*.

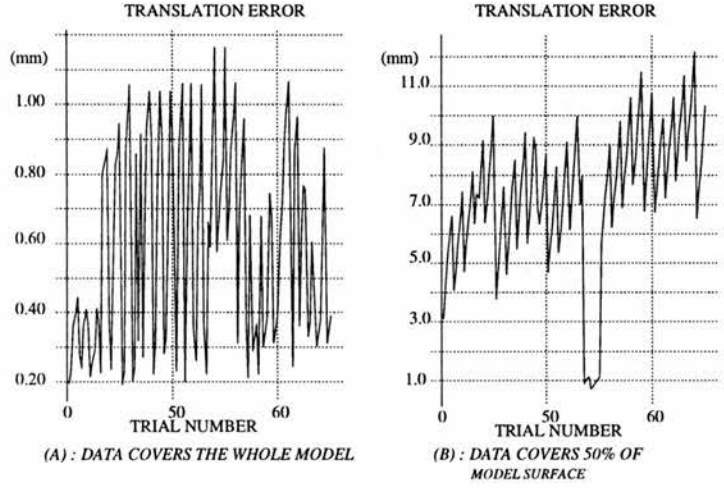


Figure 5.16: Variation of translation error ($\|\hat{\mathbf{T}} - \mathbf{T}\|$) with the value of the initial estimate of translation - Test object **1**. Initial estimates of translation vary between -2 and 2 times the dimensions of the object. Subfigure (A) covers the complete model and subfigure (B) uses data from a subset of the model.

It is clear from Figure 5.16 that the use of the ICP for inspection purposes implies the use of model and data shapes of approximately the same size and careful determination of the initial estimate of translation. We shall speak more about the use of the ICP for inspection purposes later.

The sensitivity of the ICP algorithm to the initial estimate of rotation is much bigger than its sensitivity to the initial estimate of translation. Depending on the initial estimate of rotation, the ICP algorithm might end up stuck in a local minimum very far away from the correct registration (we observed errors in the estimate of the translation

up to 100 times the dimensions of the object being registered).

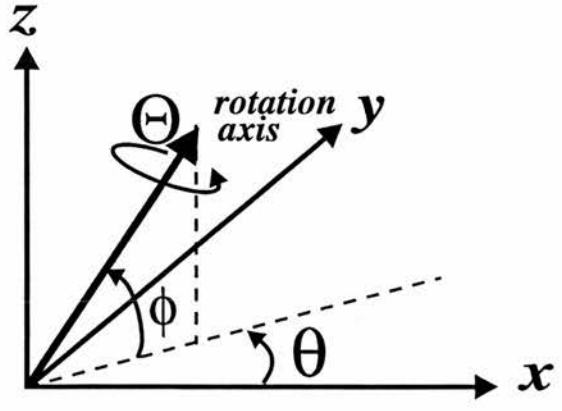


Figure 5.17: Rotation parameterisation using the angles θ, ϕ, and Θ.

The smaller sensitivity to errors in the initial estimate of translation is due to the fact that the centers of mass of the model points and the data points are superposed at the first iteration of the ICP. This fact also explains the increase in the sensitivity to errors in the initial estimate of the translation when the data points correspond just to a small fraction of the model shape (in general the error of the matching between the two centers of mass is bigger in this case).

In order to observe the variation of the performance of the ICP with the initial estimate of rotation, we proceeded with experiments in which we considered different initial estimates of the rotation $\mathbf{R}$ aligning the range images of the test objects and their corresponding B-spline models.

In our experiments we defined the different rotations using a vector corresponding to the axis of rotation (defined by the angles θ and ϕ) and the angle of rotation around this vector Θ. This means that all the rotations were parameterised using three angles θ, ϕ and Θ as illustrated in Figure 5.17.

The ICP converged to the right answer in almost all the experiments with test object **1** in which Θ was smaller than approximately *1.0 rad.* Figures 5.18 and 5.19 show the values of D_M and $\|\hat{\mathbf{T}} - \mathbf{T}\|$ obtained as a function of the angles θ and ϕ for Θ equal to *2.35 rad.* The results correspond to the same sampling grids used in Figure 5.14.

In the case of test object **2** the ICP converged to the exact answer much less often than with the test object **1**. The values of D_M and $\|\hat{\mathbf{T}} - \mathbf{T}\|$ obtained for Θ equal to

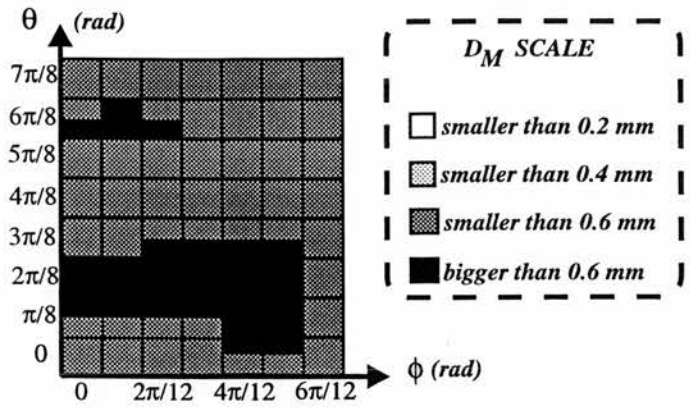


Figure 5.18: Value of D_M obtained as a function of the angles θ and ϕ for an angle Θ equal to *2.35 rad.* Results of applying the ICP to the test object **1**.

2.35 rad and for different values of θ and ϕ are shown in Figures 5.20 and 5.21.

As we can see from the figures, the convergence to the right answer corresponded to values of D_M smaller than *0.6 mm* (with test object **1**) and *0.1 mm* (with test object **2**). These numerical values were already expected and they correspond to the average distance between the range images and the models when they are perfectly aligned - see Section 5.2.

In the case of test object **1** the value of D_M corresponding to the right answer was approximately *25* % bigger than the average distance between model and data found in Section 5.2. This happened due to the presence of a few wrong correspondences that the consistency checks were not able to detect.

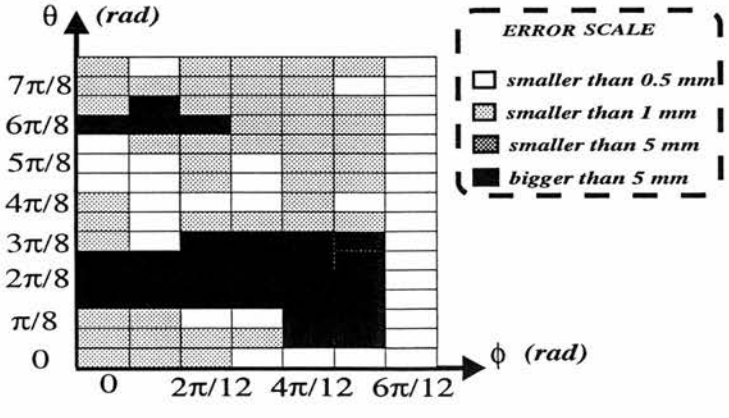


Figure 5.19: Value of $\|\mathbf{T} - \hat{\mathbf{T}}\|$ obtained as a function of the angles θ and ϕ for an angle Θ equal to *2.35 rad.* Results of applying the ICP to the test object **1**.

The graphs illustrate how the algorithm's performance is very sensitive to the initial

estimate of rotation. Furthermore, they also show that the sensitivity to the initial estimate of rotation can change drastically with the shape of the object being registered.

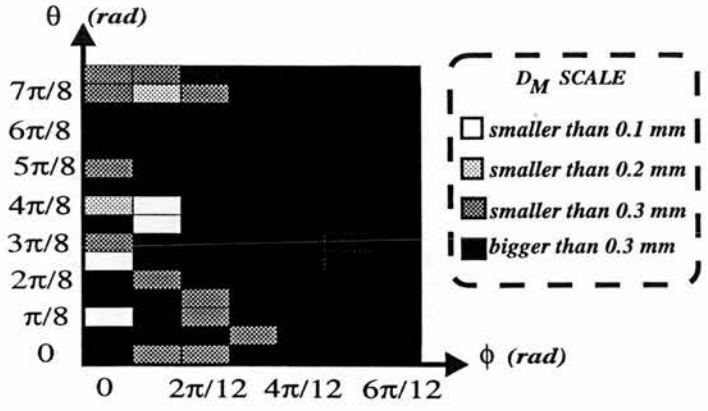


Figure 5.20: Value of D_M obtained as a function of the angles θ and ϕ for an angle Θ equal to *2.35 rad.* Results of applying the ICP to the test object **2**.

Figures 5.18 to 5.21 illustrate how test object **1** has a much smaller sensitivity to the initial estimate of rotation, *i.e.* the ICP has a much bigger convergence region in the case of test object **1**. This is probably explained by the fact that test object **1** corresponds to a surface with bigger values of curvature and, therefore, there is more information to constrain the possible solutions for the registration.

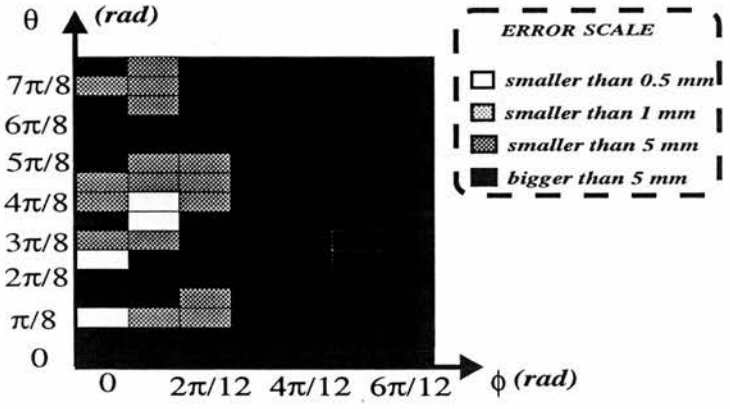


Figure 5.21: Value of $\|\mathbf{T} - \hat{\mathbf{T}}\|$ obtained as a function of the angles θ and ϕ for an angle Θ equal to *2.35 rad.* Results of applying the ICP to the test object **2**.

From all that was exposed, ICP can be used for inspection purposes according to two different approaches:

1. If we consider many runs of the modified ICP algorithm in which we start from different values of the initial estimate of pose (as suggested by [Besl & McKay 92]).

This approach implies a pre-processing stage in which simulations are used to determine a suitable set of initial guesses for the pose when the ICP is applied to a given model shape **X**.

2. If the ICP is just used to obtain a more accurate estimate of the registration between model and data. This approach can be implemented by using fixtures to constrain the pose of the surface being inspected during the data acquisition.

The choice between these two approaches depends upon the particular situation considered. The first approach will clearly demand an increase in the computational costs of the inspection. Its application is justifiable in cases where we are not interested in a big throughput rate, or we have enough computation power, or the shape of the surface is such that the ICP does not have much sensitivity to the initial guess of pose.

The use of the first approach is also possible if we consider more efficient implementations of the ICP algorithm with smaller on-line computational time. For instance, the calculation off-line of Voronoi diagrams (as suggested in [Bentley *et al.* 80]) can be used to reduce the computational time during the coarse approximation phase described in the modified ICP algorithm of Section 5.4.

The second approach can be used in situations in which it is possible to obtain an initial estimate of the pose of the surface being inspected. This approach has the advantage of not being very expensive as it is not necessary to obtain a very accurate, and consequently very expensive, initial estimate of the pose. If we consider the use of fixtures, for instance, they need only to be accurate enough to guarantee that the ICP starts from a reasonable initial estimate of pose.

The accuracy of the initial estimate of the pose necessary to guarantee the convergence of the algorithm using the second approach can be determined by using the model shape in experiments similar to the ones corresponding to Figures 5.18 to 5.21.

5.6 Conclusions

In this chapter we discussed the use of the ICP algorithm for the inspection of sculptured surfaces.

The maximum possible accuracy in registration estimate using corresponding pairs of 3D model points and 3D data points was determined as a function of the sensor errors.

A modified version of the ICP that uses a multiscale approach to increase the efficiency and consistency checks to make the method more robust to correspondence errors and outliers was presented.

The use of the ICP algorithm for inspection purposes was analysed.

Results illustrating the performance of the modified ICP with real and synthetically created data were shown to illustrate the accuracy of the method, as well as the problem of convergence to local minima.

Chapter 6

Detection and measurement of waviness errors

This chapter presents a method for the detection, amplitude measurement and localisation of periodic deformations on sculptured surfaces. A preliminary description of the method can be seen in [Bispo & Fisher 94a].

Section 6.1 presents the motivation for developing a special algorithm for detecting periodic deformations. Section 6.2 gives an overview of our method. Reliability, sensitivity and localisation characteristics of the method are discussed in Sections 6.3 to 6.5. Simulation results are presented in Section 6.6. Section 6.7 presents our final conclusions.

6.1 Motivation

The control of the shape of sculptured surfaces plays an essential role in the design and manufacture of many modern objects such as videos, audio, cars, aeroplanes, etc. It is through the careful design and control of shape in the manufacturing process that some important attributes such as aesthetic appearance, impact resistance, ease of cleaning, durability, aerodynamic behaviour, etc., are obtained.

Volume production methods of sculptured surfaces such as injection molding, die casting, forging, stamping and forming require the initial manufacture of molds and dies which require highly polished and hardened surfaces to guarantee that the desired precision in shape is achieved. This makes the production of molds and dies very ex-

pensive and time-consuming, as much of the process is performed manually by skilled technicians.

Most of the production time in the manufacturing of molds is consumed by surface finishing operations that occur after rough machining. Finishing includes grinding and polishing operations to eliminate surface errors that include [Dinauer *et al.* 94]:

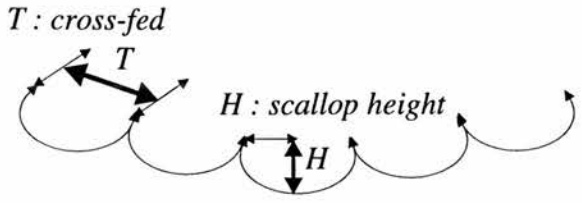
- **shape errors** - caused by undercutting or overcutting during machining, machine tool positioning inaccuracies, machine and tooling deflection under load,
- **waviness errors** - scalloped or cusped surfaces left by spherical and ball-nose cuttings, and
- **roughness errors** - which can be caused by chatter and traverse tools marks left on the surface.

Traditionally, the inspection of shape in this process is done either by using hard templates or coordinate measuring machines. Both methods are not ideal because they do not allow a complete evaluation of shape. In this chapter we address the use of dense range data for the inspection of sculptured surfaces. The idea is to use the dense grid of measured points produced by range sensors for the evaluation of shape as a whole.

The production of sculptured surfaces involves many milling and grinding operations whose objective is to achieve the desired shape quality - see Figure 6.1. The control of waviness errors during these operations is essential to the whole manufacturing process; *i.e.* the fast and reliable measurement of waviness errors is a key element in the manufacturing process. Furthermore, in many practical situations the measurement of waviness errors has to be fast enough to allow on-line control of milling and grinding operations.

Periodic deformations can also appear in the production of rolled metal [Smith 94]. Regularly spaced sequences of features can appear due to imperfections of the rollers and metal processing.

All these facts motivated us to design a particular method for the detection and meas-



FINISHING PASS	T	H
MILL 1	4.360 mm	0.500 mm
MILL 2	2.760 mm	0.200 mm
GRIND 1	1.130 mm	0.050 mm
GRIND 2	0.710 mm	0.020 mm
GRIND 3	0.500 mm	0.010 mm
GRIND 4	0.360 mm	0.005 mm

Figure 6.1: Typical shape and amplitude of waviness errors produced during a typical series of milling and grinding operations on a sculptured surface.

urement of periodic deformations on the surface of sculptured surfaces. Our main objective was the implementation of a fast and robust method that could be used in the on-line control of the milling and grinding operations.

In order to achieve our objective we decided to make use of the periodicity of the deformations to facilitate their detection and measurement. The periodic nature of the deformations causes a concentration of deformation energy at some particular points in the space of frequencies. This property suggests the use of the 2D Fourier transform for detecting the deformations.

6.2 Method description

After registration, if we ignore the registration and mapping errors, the residual image $\delta_r(x, y)$ (that corresponds to the difference between data points and model points after registration) can be modelled by Equation (6.2.1). In this equation $\epsilon_d(.)$ corresponds to the periodic deformation we want to detect and $\eta(.)$ to a random process.

$$\delta_r(x, y) = \epsilon_d(x, y) + \eta(x, y) \tag{6.2.1}$$

$$\mathbf{E}[\eta(x, y)] = 0 \tag{6.2.2}$$

$$\mathbf{E}[\eta(x,y)\eta(x,y)] = {\sigma_\eta}^2 \quad (6.2.3)$$

$$\mathbf{E}[\eta(x,y)\eta(x',y')] = 0 \quad (x \neq x' \ \ or \ \ y \neq y') \quad (6.2.4)$$

$$x = \{0,1,...,L-1\} \quad (6.2.5)$$

$$y = \{0,1,...,P-1\} \quad (6.2.6)$$

where:

- $\mathbf{E}[.]$ is the expectation operator.
- $\delta_r(x,y)$ is the value of the residual image in the pixel (x,y).
- L and P are the dimensions of the residual image in the x and y direction respectively.

The statistical model of the random process η in the residual image depends on many factors such as:

- the rigid transformation aligning model to the data,
- the errors in the determination of this rigid transformation,
- the errors in the definition of the mapping function between the data points and the model,
- the statistical model of the random errors of the sensor and
- the statistical model of the random errors of the manufacturing process.

In this chapter we will assume that the random errors in the manufacturing process are much smaller than the sensor random errors, and that the sensor errors are spatially uncorrelated and normally distributed. This means that $\eta(.)$ is due mainly to the errors in the sensor. Throughout this chapter we will assume that σ_η is equal to σ_S (that is the standard deviation of the error in the range measurement).

Our inspection strategy consists of analysing the DFT (2D Discrete Fourier Transform) of the residual image in sub-windows (that we will call observation windows) covering the whole residual image. The residual image is broken into observation windows in

order to increase the sensitivity of the method and allow the detection of periodic deformations that occupy small areas in the residual image.

The following procedure is applied to each observation window:

1. Calculation of the DFT of the residual image that covers the observation window.
2. Thresholding of the modulus of the DFT of the residual calculated in step 1. Peaks in the frequency domain bigger than a threshold value τ_1 indicate the presence of periodic deformations.
3. If a deformation is detected, the region of the deformation within the observation window is found by applying the inverse DFT to the selected peaks of the original DFT, and then keeping deformations whose values are bigger than a threshold value τ_2. This procedure also produces the amplitude of the detected deformations.

After the whole residual image is processed, a global estimate of the deformation region in the residual image is obtained by the union of all the deformed regions found in the observation windows.

In the algorithm described above, the determination of the threshold values τ_1 and τ_2 is directly related to many factors:

- the shape of the periodic deformation;
- the minimum amplitude and area occupied by periodic deformation we want to be able detect;
- the statistical model of $\eta(.)$;
- the maximum acceptable probability of false alarm (P_f^{dft});
- the maximum acceptable probability of not detecting a given deformation (P_n^{dft});
- the maximum acceptable probability of a point not belonging to the deformation being classified as in the deformation (P_1);

- the maximum acceptable probability of a point belonging to the deformation not being detected (P_2).

The registration errors and the model of the random errors in the manufacturing process are also relevant when they are not much smaller than the sensor errors, contrary to the assumption we made earlier.

In the following sections we discuss the relation of the threshold values τ_1 and τ_2 to all the factors cited previously.

It is important to notice that a significant limitation in the detection ability is the spatial frequency of the deformation. In order to avoid aliasing problems, the spatial period of the deformation should be bigger than 2 pixels.

6.3 Thresholding the DFT of the residual

In this section we focus on the determination of the threshold value τ_1 as a function of the probability of false alarm P_f^{dft} and the probability of not detecting a given deformation P_n^{dft}.

6.3.1 Probability of false alarm

Without any loss of generality, we will consider square observation windows of size N. The DFT of $\delta_r(x,y)$, which we will call $R(u,v)$, in the observation window is given by:

$$R(u,v) = \frac{1}{N} \sum_{x=0}^{x=N-1} \sum_{y=0}^{y=N-1} \delta_r(x,y) {W_N}^{xu+yv} \tag{6.3.7}$$

$$W_N = e^{\frac{-2\pi j}{N}} \tag{6.3.8}$$

$$u = \{0,1,...,N-1\} \tag{6.3.9}$$

$$v = \{0,1,...,N-1\} \tag{6.3.10}$$

In this case we are interested in the probability that the modulus of $R(u,v)$ is bigger than the threshold value τ_1, when there is not any deformation in the residual, *i.e.*:

$$\delta_r(x,y) = \eta(x,y) \tag{6.3.11}$$

$$R(u,v) = Re(u,v) + jIm(u,v) \tag{6.3.12}$$

$$Re(u,v) = \frac{1}{N}\sum_{x=0}^{x=N-1}\sum_{y=0}^{y=N-1}\eta(x,y)cos(\frac{\pi(xu+yv)}{N}) \quad (6.3.13)$$

$$Im(u,v) = \frac{1}{N}\sum_{x=0}^{x=N-1}\sum_{y=0}^{y=N-1}\eta(x,y)sin(\frac{-\pi(xu+yv)}{N}) \quad (6.3.14)$$

As we assumed $\eta(.)$ to be normally distributed, $R(u,v)$ and $Im(u,v)$ are also normally distributed and, for N odd:

$$\mathbf{E}[Re(u,v)] = 0 \quad (6.3.15)$$

$$\mathbf{E}[Re(u,v)^2] = \begin{cases} 1 & \text{u=0 and v=0} \\ \frac{{\sigma_\eta}^2}{2} & \text{otherwise} \end{cases} \quad (6.3.16)$$

$$\mathbf{E}[Im(u,v)] = 0 \quad (6.3.17)$$

$$\mathbf{E}[Im(u,v)^2] = \begin{cases} 0 & \text{u=0 and v=0} \\ \frac{{\sigma_\eta}^2}{2} & \text{otherwise} \end{cases} \quad (6.3.18)$$

$$\mathbf{E}[Im(u,v)R(u,v)] = 0 \quad (6.3.19)$$

For calculating the probability of false alarm P_f^{dft} let us consider the random variable $V(.)$ given by:

$$V(u,v) = Re(u,v)^2 + Im(u,v)^2 \quad (6.3.20)$$

There will be a false alarm condition when the modulus of $R(u,v)$ is bigger than τ_1, *i.e.* when $V(u,v)$ is bigger than ${\tau_1}^2$. Therefore, the probability of false alarm P_f^{dft} can be found by integrating the joint distribution of the random variables $Re(u,v)$ and $Im(u,v)$ in a disc of radius τ_1. As $Re(u,v)$ and $Im(u,v)$ are both normal distributed, we obtain:

$$P_f^{dft} = 1 - \frac{1}{\pi{\sigma_\eta}^2}\int_{-\tau_1}^{\tau_1}\int_{-\sqrt{{\tau_1}^2-x^2}}^{\sqrt{{\tau_1}^2-x^2}} e^{\frac{-(x^2+y^2)}{{\sigma_\eta}^2}}\,dxdy \quad (6.3.21)$$

The integral in Equation (6.3.21) can be easily solved and, for $u \neq 0$ and $v \neq 0$:

$$P_f^{dft} = e^{-(\frac{\tau_1}{\sigma_\eta})^2} \quad (6.3.22)$$

This result is very intuitive, it shows that the probability of false alarm P_f^{dft} depends only on the average energy of the random process η (${\sigma_\eta}^2$) and the minimum energy that a spectral component of the deformation must have to be detected (${\tau_1}^2$).

6.3.2 Probability of not detecting a deformation

In this section we want to find the relation between the probability P_n^{dft} of not detecting a given deformation and the threshold value τ_1 or, in other words, we want to find the

probability of the modulus of the DFT of the residual image $R(u,v)$ being smaller than our threshold when we have a deformation $\epsilon_d(.)$.

As in the previous section, $Re(u,v)$ and $Im(u,v)$ are gaussian variables with the same variances as those given in Equations (6.3.16) and (6.3.18). However, if $E(u,v)$ is the DFT of $\epsilon_d(.)$ in the observation window, the means of $Re(u,v)$ and $Im(u,v)$ are:

$$\mathbf{E}[Re(u,v)] = m_R \tag{6.3.23}$$

$$\mathbf{E}[Im(u,v)] = m_I \tag{6.3.24}$$

$$E(u,v) = m_R + jm_I \tag{6.3.25}$$

$$|E(u,v)| = M = \sqrt{{m_R}^2 + {m_I}^2} \tag{6.3.26}$$

Now we are going to find the probability P_n^{dft} of a spectral component of $E(.)$ of modulus M not being detected when we use a threshold value τ_1. It is important to notice that spectral components with modulus bigger than M will have probabilities smaller than P_n^{dft} of not being detected.

As before, the probability of not detecting a deformation can be found by integrating the joint distribution of $Re(u,v)$ and $Im(u,v)$ in a disc of radius τ_1:

$$P_n^{dft} = \frac{1}{\pi {\sigma_\eta}^2} \int_{-\tau_1}^{\tau_1} \int_{-\sqrt{{\tau_1}^2-x^2}}^{\sqrt{{\tau_1}^2-x^2}} e^{-\frac{(x-m_R)^2+(x-m_I)^2}{{\sigma_\eta}^2}} dxdy \tag{6.3.27}$$

This integral does not have an analytical solution, but a few manipulations give us an easier equation for numerical integration:

$$P_n^{dft} = \frac{1}{\pi} \int_{\frac{1-\alpha}{\sigma_\eta}M}^{\frac{1+\alpha}{\sigma_\eta}M} xe^{-x^2} \arccos \frac{(1-\alpha^2)M^2 + x^2{\sigma_\eta}^2}{2x\sigma_\eta M} dx \tag{6.3.28}$$

$$\alpha = \frac{\tau_1}{M} \quad 0 < \alpha < 1 \tag{6.3.29}$$

It is important to notice that for $\tau_1 = M$ we have P_n^{dft} equal to 0.5, and P_n^{dft} is monotonic ascending with τ_1. As desirable values of P_n^{dft} should be much smaller than 0.5, we will just consider values of τ_1 smaller than M. Figure 6.2 show the variation of P_n^{dft} as a function of α for different values of ratio $\frac{M}{\sigma_\eta}$.

6.3.3 Determining τ_1

Equations (6.3.22) and (6.3.28) show that the determination of τ_1 involves a trade-off between the probability of false alarm and the probability of not detecting a deforma-

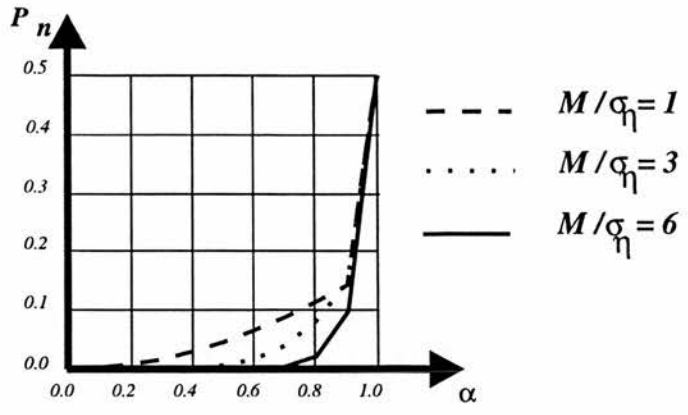


Figure 6.2: Variation of P_n^{dft} with α and $\frac{M}{\sigma_\eta}$.

tion, *i.e.* as in Chapter 4, there is a trade-off between the reliability and the sensitivity of the inspection procedure.

The bigger the value of τ_1, the smaller the chance of false alarm; but on other hand, when the value of τ_1 increases, the probability of not detecting a real periodic deformation also increases. This fact is clearly illustrated in Figures 6.2 and 6.3 which show the variation of P_f^{dft} and P_n^{dft} as a function of α and $\frac{M}{\sigma_\eta}$. The ratio $\frac{M}{\sigma_\eta}$ represents the square root of the signal to noise ratio in the point (u, v) of the spectrum.

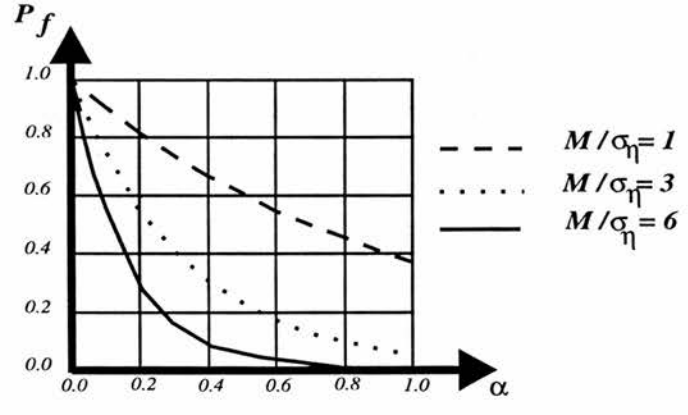


Figure 6.3: Variation of P_f^{dft} with $\frac{M}{\sigma_\eta}$.

In order to reconstruct the deformation, we need to recover all the main spectral components of $E(u, v)$. Therefore, the threshold τ_1 is determined using the value of M corresponding to the spectral component of $E(u, v)$ with smallest modulus. It follows therefore that the probability of not detecting a given deformation during the inspection procedure should be smaller than P_n^{dft} as given by Equation (6.3.28).

In the determination of τ_1 we assumed that we have an estimate of an upper limit for σ_η, which is obtained by modelling the sensor accuracy. This value is used together with the desired value of the probability of false alarm P_f^{dft} and Equation (6.3.22) to determine the threshold value τ_1. After that, the maximum acceptable value of P_n^{dft} is used together with Equation (6.3.28) to determine the minimum value of modulus M that will have probability P_n^{dft} of not being detected. This value is obtained by solving Equation (6.3.28) numerically. The numerical solution is obtained by using linear search and adaptive recursive Simpson's rule for numerical integration.

Once the minimum value of M is found, this value is used to determine the minimum amplitude and size of the deformation that can be detected with probabilities P_f^{dft} and P_n^{dft} (more details about this procedure are given in the next section).

It is not possible to choose independently a reliability (the value of P_f^{dft}) and sensitivity (minimum amplitude and size of deformation detectable with a probability $1 - P_n^{dft}$). The reliability and sensitivity of the method in this step are limited by the amount of noise in the residual image (σ_η) and the shape of the periodic deformation. Larger values of reliability will be obtained only by reducing the sensitivity.

As a design option, we chose to determine τ_1 for a given reliability and than evaluate the corresponding sensitivity. This choice is justified by the fact that this method presents very good sensitivity . Also any amplitude of deformation can be detected as long as the deformation occupies a minimum area in the residual image. The determination of the minimum amplitude and size of deformation detectable with probability $1 - P_n^{dft}$ for a given reliability is discussed in Section 6.4.

6.4 Sensitivity: minimum amplitude and size of deformation detectable

Using Equations (6.3.22) and (6.3.28) we can determine the minimum value of the modulus of $E(u, v) = M$ that can be detected with probabilities P_f^{dft} and P_n^{dft} of error. The values of M and P_n^{dft} are in themselves a measurement of the sensitivity of the method for a given reliability, but they do not give much useful information. We want to determine more useful information such as:

- the minimum area that a periodic deformation must occupy to be detectable with probability P_n^{dft},
- the minimum amplitude that a periodic deformation must have to be detectable with probability P_n^{dft},
- and the ideal size of observation window to use.

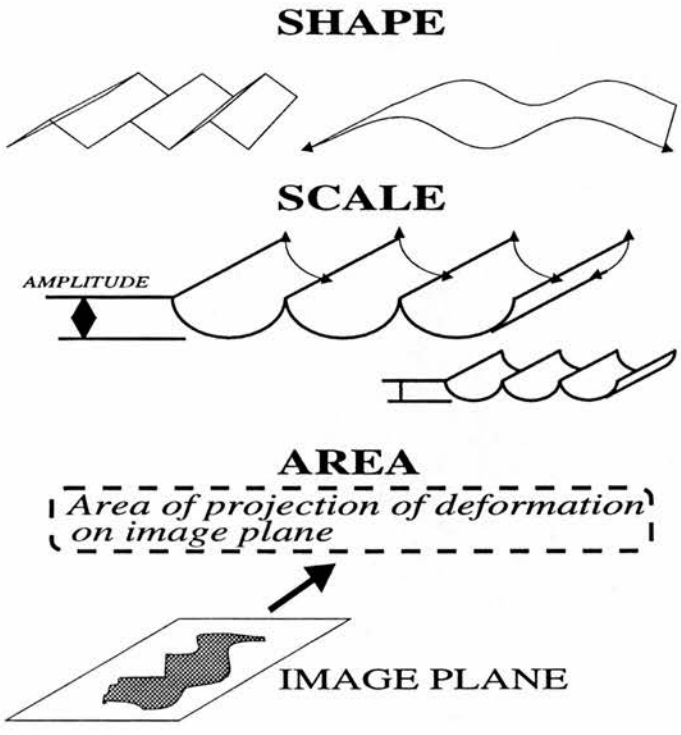


Figure 6.4: Illustration of shape, scale (amplitude) and area of periodic deformations.

As we observed before, the value of M is related to the minimum energy a spectral component of the deformation must have in order to be detected with probability P_n^{dft}. This value depends basically on three factors (see Figure 6.4):

- *shape of deformation*: the geometric shape of the periodic deformation;
- *scale*: the size of the deformation. We will measure the size using the length of a characteristic dimension in the deformation that we will call amplitude;
- *area*: area occupied by the projection of the deformation on the image plane of the range image.

In our case we are interested in a special shape of deformation that is characteristic of the waviness errors and presents a shape similar to the shape shown in Figure (6.1). We will model this kind of deformation as a set of parallel ridges as shown in Figure (6.5). The model uses seven parameters to control the amplitude (H), the period (D_1) and (D_2) , the orientation (γ), position (x_0, y_0) and area of the deformation (l^2).

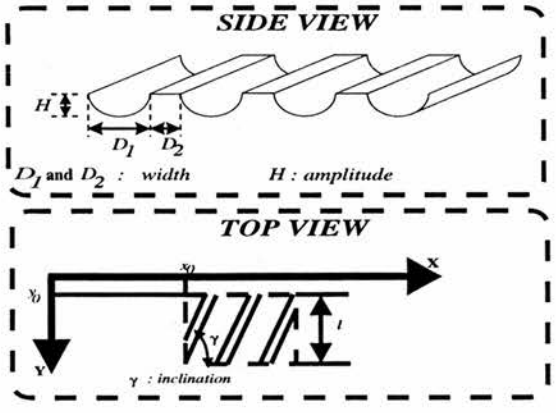


Figure 6.5: The geometric model adopted for waviness errors.

The minimum size l^2 and amplitude H that a ridge deformation must have to allow its recovery from the spectrum components of the deformation with modulus bigger than M is shown in Figure 6.6. Given a value of M necessary to reach the desired values of P_f^{dft} and P_n^{dft}, this graph allows us to determine:

- either the minimum area l^2 the deformation must have when we know the minimum amplitude H of the deformation we want to detect or
- the minimum amplitude of deformation H_{min} that we can detect when we know the minimum area l^2 of the deformation.

The graph was obtained by simulating ridge deformations with different values of parameters and observing the spectrum $E(u, v)$ of these deformations to determine the corresponding value of M. The aim was to determine which part of the spectrum of $E(u, v)$ was necessary for reconstructing the deformation.

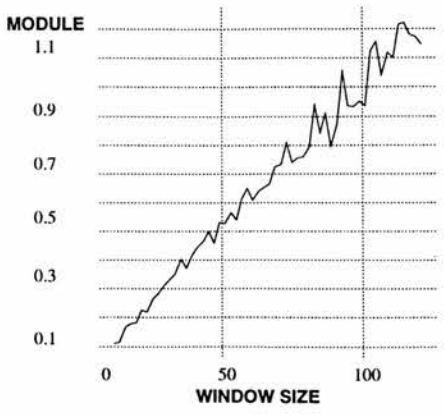


Figure 6.6: Relation between the ratio M and the minimum size l of deformation. D_1, D_2 from 1.5 mm to 15 mm, γ from 0 to 90 degrees and $H_{min} = 0.1$ mm.

6.4.1 The observation window

So far, we have discussed the determination of the threshold value τ_1 as a function of a given reliability, as well the sensitivity of the inspection procedure for a given threshold value. Another very important element in the method is the determination of the size of the observation window.

From now on we shall refer to the length (l) of the side of the square delimiting the observation window as the *size* of it.

Clearly, the size of the window should be big enough to cover at least one period of the deformation equal to $D_1 + D_2$. Also, for a given reliability and minimum value of deformation amplitude to detect, using Equations (6.3.22), (6.3.28) and Figure 6.6, we can determine minimum size of window the deformation must cover to be detected. We shall call this value L_d. Therefore, the size of the observation window should be larger than W_d given by:

$$W_d = max\{L_d, D_1 + D_2\} \tag{6.4.30}$$

We cannot use observation windows of size W_d because we do not know the exact position of the deformation within the residual image, and we must have at least a window of size W_d inside of the observation window covered by the deformation. To solve this problem, we propose to use observation windows of size W_o:

$$W_o = 2\beta W_d \tag{6.4.31}$$

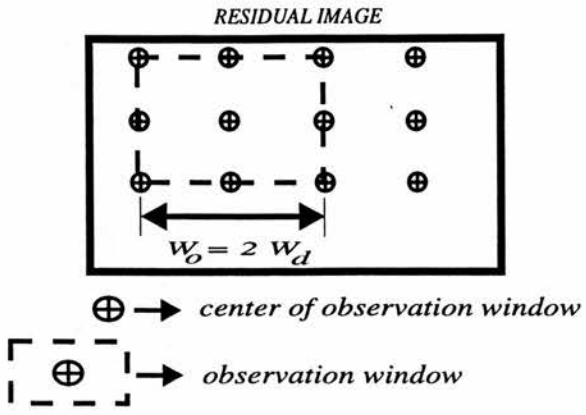


Figure 6.7: Position of observation windows inside of the residual image.

The idea is to use observation windows of size equal to at least twice W_d such that the center of adjacent observation windows are at distance W_d. Figure 6.7 illustrates the inspection procedure. Because of this, adjacent observation windows will overlap and the residual image will actually be tested more than once. This procedure is necessary to make sure that a deformation covering a square window of size W_d will be always be detected whatever its position is within the residual image. In Equation (6.4.31), β is a security factor, usually set empirically to 1.0 or 1.2.

6.5 Reconstructing the deformation inside the observation window

When a deformation is detected inside an observation window the reconstruction of the deformation inside the window is performed by applying the inverse DFT to the selected peaks of $E(u, v)$ and then thresholding the results of the inverse DFT. The objective of this process is twofold: to localise the deformation within the deformation window and to estimate the amplitude of the deformation.

The threshold value τ_2 used during this process depends on many factors:

- the statistical model of the random process $\eta(.)$,
- the probability P_1 of a point not belonging to the deformation being detected,
- the probability P_2 of a point in the deformation not being detected.

In order to relate τ_2 to P_1 and P_2 let us start by considering the reconstructed residual $\hat{\delta_r}(x,y)$ produced by the inverse DFT applied to the selected peaks of $E(u,v)$:

$$\hat{\delta_r}(x,y) = \frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} K(u,v)(E(u,v)+N(u,v))W_N{}^{-xu-yv} \quad (6.5.32)$$

$$\hat{\delta_r}(x,y) = \hat{\epsilon_d}(x,y)+\hat{\eta}(x,y) \quad (6.5.33)$$

$$\hat{\epsilon_d}(x,y) = \frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} K(u,v)E(u,v)W_N{}^{-xu-yv} \quad (6.5.34)$$

$$\hat{\eta}(x,y) = \frac{1}{N}\sum_{u=0}^{u=N-1}\sum_{v=0}^{v=N-1} K(u,v)N(u,v)W_N{}^{-xu-yv} \quad (6.5.35)$$

$$K(u,v) = \begin{cases} 1 & |E(u,v)+N(u,v)| \geq \tau_1 \\ 0 & \text{otherwise} \end{cases} \quad (6.5.36)$$

Equation (6.5.32) shows that the reconstructed deformation is composed of two parts:

- a reconstruction of the original deformation ($\hat{\epsilon_d}(.)$) and
- a reconstruction of the original random process ($\hat{\eta}(.)$).

As $\hat{\eta}(.)$ is a linear combination of normally distributed random variables, $\hat{\eta}(.)$ is also a normally distributed random variable and:

$$\mathbf{E}[\hat{\eta}(x,y)] = 0 \quad (6.5.37)$$

$$\mathbf{E}[\hat{\eta}(x,y)\hat{\eta}(x,y)] = \frac{\sigma^2}{N^2}K_{ON} = \sigma_{\hat{\eta}}{}^2 \quad (6.5.38)$$

$$K_{ON} = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} K(u,v) \quad (6.5.39)$$

Typically, the value of K_{ON} will never be bigger than 36, therefore considering Equation (6.5.38) a reasonable upper limit to the standard deviation $\sigma_{\hat{\eta}}$ is:

$$\sigma_{\hat{\eta}} \leq 6\frac{\sigma_\eta}{N} \quad (6.5.40)$$

The probability P_1, of a point (x,y) not belonging to the deformation being classified as belonging to the deformed region corresponds to the probability of $\hat{\eta}(x,y)$ being bigger than the threshold value τ_2, *i.e.*:

$$P_1 = 1-\frac{1}{\sqrt{2\pi}\sigma_{\hat{\eta}}}\int_{-\tau_2}^{\tau_2} e^{\frac{-x^2}{\sigma_{\hat{\eta}}{}^2}}dx \quad (6.5.41)$$

The probability P_2, of a point (x,y) belonging to the deformation not being detected depends on the value of $\rho = \hat{\epsilon_d}(x,y)$ that is the value of the reconstructed deformation

in the point (x, y). This means that different points in the deformation will have different probabilities of being detected. For instance, points in which ρ is equal to zero will have a probability $1 - P_1$ of not being detected. Also the bigger the absolute value of ρ the smaller P_2 will be.

The probability P_2 corresponds to the probability of $|\rho + \hat{\eta}(x, y)| < \tau_2$. Therefore:

$$P_2(\rho) = \frac{1}{\sqrt{2\pi}\sigma_{\hat{\eta}}} \int_{-\tau_2-\rho}^{\tau_2-\rho} e^{\frac{-x^2}{\sigma_{\hat{\eta}}^2}} dx \tag{6.5.42}$$

A lower limit for P_2 can be established by considering ρ equal to the minimum amplitude of deformation we want to detect. This is a very conservative limit for P_2, being based on the very worst possible case.

In order to determine the value of τ_2 we considered principally the value of P_1. This choice is justified by the fact that P_2 will vary from $1 - P_1$ up to some lower limit. Also, we considered that the most important factor was to guarantee that practically no point that did not belong to the deformation would be detected. As a rule of thumb we used:

$$\tau_2 = \frac{H_{min}}{2} \tag{6.5.43}$$

where: H_{min} is the minimum amplitude of deformation we want to detect.

This rule of thumb is based on the idea that H_{min} must be bigger than $6\sigma_{\hat{\eta}}$ for a reconstruction of the deformation with an acceptable accuracy. The definition of τ_2 in Equation (6.5.43) therefore produces a good balance between the value of P_1 and the lower limit of P_2.

Exact determination of the deformed region is not possible. However, a close approximation to the deformation region can be obtained as well as the amplitude of the deformation region, because a good percentage of the deformed points will be detected and observation windows will typically have at least 1000 pixels.

To calculate the percentage of points detected in the deformed region we will use the function $P_2(\rho)$ defined by Equation (6.5.42). If we consider an observation window of size $W_o = l$ and pixel area equal to $\Delta A = s_x s_y$, the percentage of points detected, Av, is:

$$Av = \frac{1}{l^2} \int_{\rho_{min}}^{\rho_{max}} (1 - P_2(\rho)) N_\rho d\rho \tag{6.5.44}$$

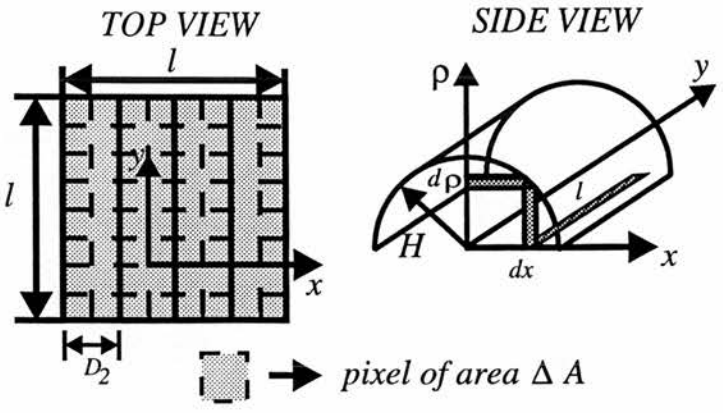


Figure 6.8: Illustration of ridge deformation inside the observation window.

where $N_\rho d\rho$ is the number of points in the infinitesimal deformed region for which $\hat{\epsilon_d}(x,y) = \rho$.

For a ridge deformation with D_2 and γ both equal to zero (see Figure 6.5) we have (see Figure 6.8):

$$\rho = H\sqrt{(\frac{2x}{D_1}^2 - 1)} \tag{6.5.45}$$

$$N_\rho d\rho = \frac{dx}{s_x} \tag{6.5.46}$$

$$\rho_{min} = 0 \tag{6.5.47}$$

$$\rho_{max} = H \tag{6.5.48}$$

Therefore, Equation (6.5.44) becomes:

$$Av = \frac{2}{D_1}\int_{\frac{D_1}{2}}^{0}(1 - P_2(H\sqrt{\frac{2x}{D_1}^2 - 1}))dx \tag{6.5.49}$$

Figure 6.9 shows the variation of P_2 with ρ for typical values of H_{min} and $\sigma_\eta = 0.1$ *mm*. The graph shows the division of the deformed points in three regions of points:

- points with probability close to one of not being detected (near $\rho = 0$),
- points with probability close to zero of not being detected (near $\rho = H_{min}$) and
- points belonging to a transition region between the two previous regions (near $\rho = 0.5H_{min}$).

The bigger the value of H_{min} in relation to σ_η the smaller and more abrupt is the transition region. This happens because when the ratio $\frac{H_{min}}{\sigma}$ increases the influence of the reconstructed random process $\hat{\eta}(.)$ decreases.

As one could expect, the value of Av in this case is equal to 0.5. This value does not depend on particular values of H_{min} and D_1 for a given value of σ_η. Also, the value of Av is very robust to variations in σ_η for observation windows with W_o equal to 100 pixels or bigger. This fact can be understood by looking at Equation (6.5.40) that shows how only very big variations in σ_η will cause considerable variations in $\sigma_{\hat{\eta}}$.

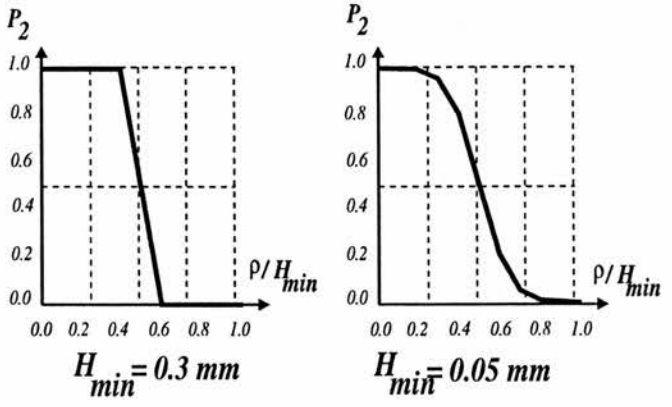


Figure 6.9: Variation of P_2 with ρ for $\sigma_\eta = 0.1mm$, $W_o = 100$ and $\tau_2 = \frac{H_{min}}{2}$.

6.6 Simulation Results

In order to test the inspection procedure we conducted simulation tests in which we adopted the following process:

1. Capture of a range image of the test object.
2. Addition of a simulated periodic deformation, as in Figure 6.5, to the range image.
3. Registration between corrupted range image and model of test object using the modified version of the ICP algorithm described in Chapter 4.
4. Application of inspection procedure for detecting, localising and measuring deformation in range image.

As before we used Object 1 and Object 2 as test objects (see Chapter 4) and their images were produced using the Edinburgh Laser Striper which has σ_S equal to *0.1 mm*. In the case of Object 2, as explained in Chapter 4, we considered σ_S equal to *0.4 mm*. The inspection procedure was implemented using MATLAB.

As one could expect, because we are registering a perfect and a deformed shape, there is a deterioration in the performance of the registration. The amount of the deterioration depends on the amplitude and area of the deformation. However, the algorithm copes well with the typical amplitudes of deformation shown in Figure 6.1 and the registration is good enough for making possible the application of the inspection procedure.

To the range image of Object 1 we added a deformation, as in Figure 6.5, with H = *0.1 mm*, D_1 = *2.5 mm*, D_2 = *0.0 mm*, $\gamma = 90°$ and l = *100 pixels*. This deformation was added to the portions of the object in the upper left corner of the range image. The residual image obtained after registration between model and the corrupted range image is shown in Figure 6.10. The DFT of the residual in an observation window over the deformation is shown in Figure 6.11.

Considering $\sigma_\eta = 0.1mm$ and $P_f^{dft} = P_n^{dft} = 0.0001$ from Equations (6.3.22) and (6.3.28) we obtain: $\tau_1 = 0.31$ and $M = 0.55$. If we consider $W_o = 2W_d$ and H_{min} equal to $0.1mm$, the graph of Figure 6.6 indicates that the minimum size of deformation detectable is W_d = 50 *pixels* which corresponds to a rectangle of *25 mm* × *50 mm* considering the scale of the range image. Following the rule of thumb we used τ_2 = *0.05 mm*.

The projection of the deformed region on the image plane found by the inspection is shown in Figure 6.12. The measured amplitude of deformation was equal to 0.12 mm. Figure 6.12 shows clearly that, as explained before, only points with ρ near H_{min} are detected.

We repeated the same procedure with Object 2 to which we added a deformation with H = *0.4 mm*, D_1 = *2.5 mm*, D_2 = *1.0 mm*, $\gamma = 45°$ and l = *100 pixels*. Again we used the same values of P_f^{dft} and P_n^{dft}, and consequently we obtained $\tau_1 = 1.2$ and $M = 2.25$. The value of W_d in this case was found by scaling the graph in Figure 6.6 and it was equal to approximately *50 pixels*. Following the rule of thumb we used

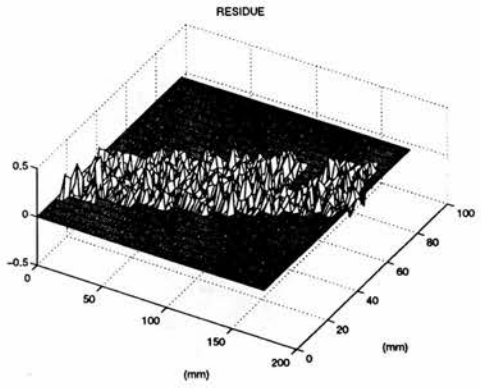


Figure 6.10: Residual image obtained after registration in the simulation with Object 1.

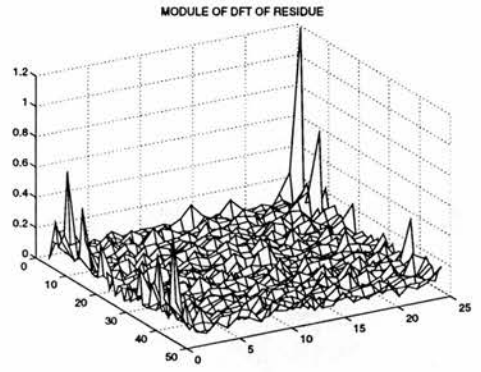


Figure 6.11: Modulus of DFT of residual image in an observation window over the deformation - Object 1.

$\tau_2 = 0.2\ mm.$

The application of the inspection procedure to Object 2 was not so successful as in the case of Object 1. In this case the algorithm detected not only the deformation that was added to the range image, but also some other periodic deformations found in the residual image. These deformations correspond to errors in the model of the Object 2 that appear due to the fitting of a spline to a surface with C_0 discontinuities.

This fact is illustrated by Figure 6.13 which shows the result of fitting a cubic spline to a step edge. As we can see in the figure, the spline approximation presents some periodic components near the C_0 discontinuity of the step edge.

Figure 6.14 shows the *residual image* in the case of Object 2. Figure 6.15 shows the DFT of the residual over the deformed region.

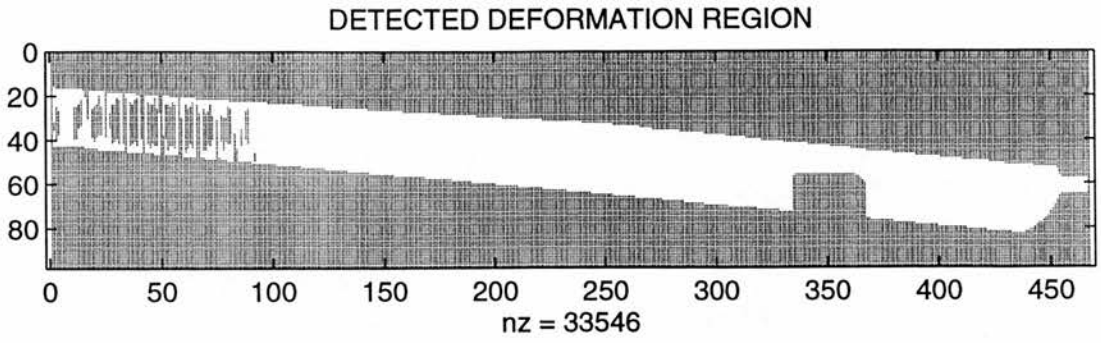


Figure 6.12: Final deformed region detected in the simulation with Object 1.

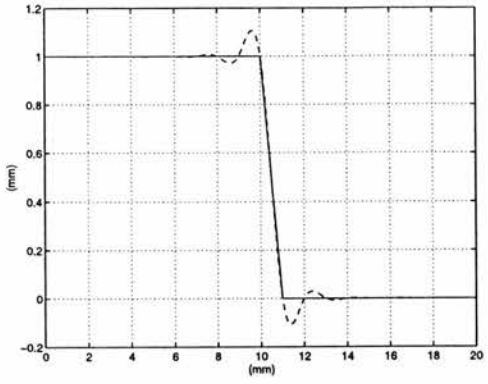


Figure 6.13: Error in the fitting of cubic spline to step edge.

Figure 6.16 shows the projection of the detected deformed region on the image plane. The figure shows clearly how the algorithm detected not only the deformations added to the upper right corner of the range image, but also the periodic components of the modelling errors near a discontinuity in the range image.

The measured value of amplitude found by the algorithm was *1.2 mm*. This value does not correspond to the amplitude of the deformation added to the range image, but to the amplitude of the deformation near a C_0 discontinuities in the range image which explains the difference to the original amplitude of the deformation (*0.4 mm*).

The results in the simulation with Object 1 showed a good agreement with the expected results. They also enhanced some of the main characteristics of the algorithm as:

- *Good sensitivity:* Even for an extremely high reliability the method is able to detect deformations with relative small amplitude and size.

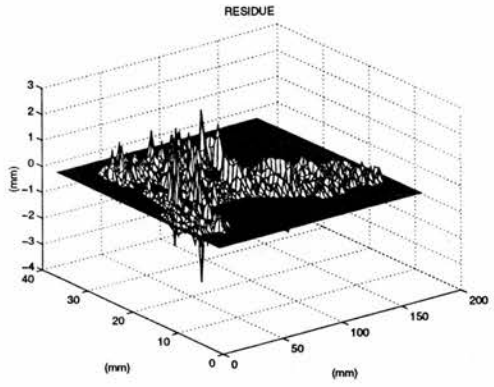


Figure 6.14: Residual image obtained after registration in the simulation with Object 2.

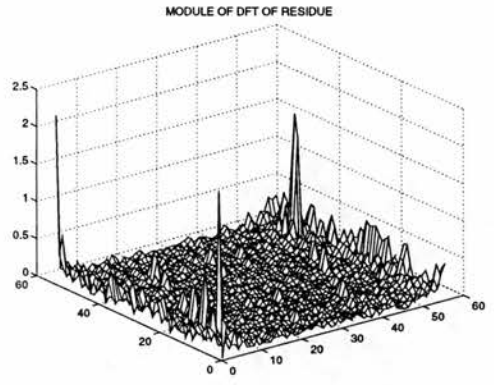


Figure 6.15: Modulus of DFT of residual image in an observation window over the deformation - Object 2.

- *Reasonable amplitude estimation:* Reasonable amplitude estimation, due mainly to the reduction of the noise effects in the inverse DFT.
- *Reasonable localisation properties:* The method produces a good estimation of the deformed region within the image, but is not capable of detecting points belonging to the deformation that have ρ near zero.

Figure 6.1 suggests that we need to measure amplitudes of the order of *0.005 mm* to inspect all the phases of the milling and grinding process. The extrapolation of the graph of Figure 6.6 leads to the conclusion that for $P_f^{dft} = P_n^{dft} = 0.0001$ and σ_S equal to *0.1 mm* we would have W_d approximately 1.100 pixels which corresponds to a square of at least *0.55* m^2!

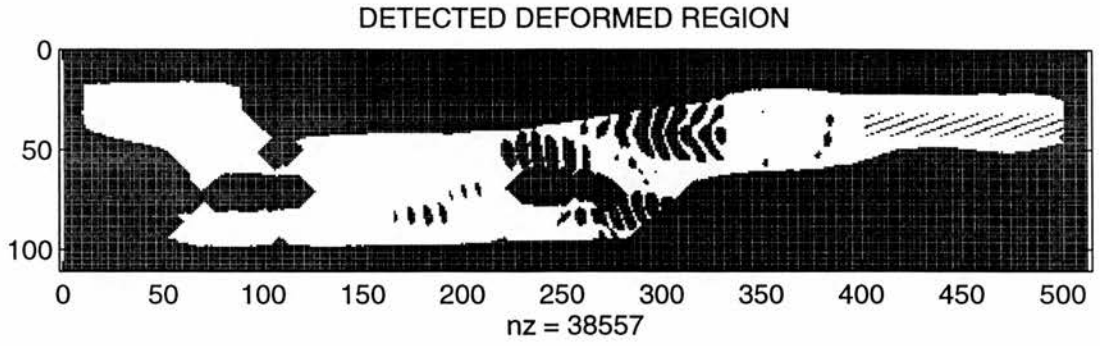


Figure 6.16: Final deformed region detected in the simulation with Object 2.

Finishing pass	W_d $(pixels)$	Max. pixel size (mm)
Mill 1	11	2.180
Mill 2	28	1.380
Grind 1	110	0.565
Grind 2	275	0.355
Grind 3	550	0.250
Grind 4	1100	0.180

Table 6.1: Minimum size of deformation $W_d = \frac{W_o}{2}$ and maximum pixel size for the inspection of milling and grinding processes. $P_f^{dft} = P_n^{dft} = 0.0001$ and $\sigma_S = 0.1\ mm$.

This shows that for the inspection of waviness errors in all the milling and grinding phases of the production process we need a laser striper at least ten times more precise (σ_S equal to $0.01mm$). Also, the cross-feed values shown in Figure 6.1 suggest that we need pixels with size of at most 0.18 mm to avoid aliasing.

Table 6.1 shows the value of $W_d = \frac{W_o}{2}$ and maximum pixel size for the inspection and measurement of waviness errors in the milling and grinding phases enumerated in Figure 6.1. These numbers were obtained by considering a sensor with $\sigma_S = 0.1\ mm$ and the values of P_f^{dft} and P_n^{dft} used in the simulation.

6.7 Conclusions

We have presented an inspection procedure for the detection and measurement of waviness errors produced on the milling and grinding operations of sculptured surfaces. The reliability, sensitivity and localisation properties of the algorithm as a function of

the design parameters of the inspection procedure were discussed. Simulation results illustrating the algorithm application were presented.

Chapter 7

Conclusions

In this chapter we present our final comments concerning our investigation on the automatic model-based inspection of mechanical parts using dense range data.

Section 7.1 introduces the chapter by comparing our initial objectives with the results of our research. Sections 7.2 to 7.4 discuss the results obtained in Chapters 4 to 6, as well as suggest possible directions for further research. Section 7.5 summarises the research done in this thesis and highlights the original contribution of the research.

7.1 Introduction

As we said in Chapter 1, the main objective of this thesis was the study of some important issues that we summarised using four questions:

1. Which types of inspection are possible to carry out using range data ?
2. What accuracy and precision are achievable given the accuracy and precision of the original range data ?
3. What is the best way of modelling the objects for inspection using range data ?
4. How to use the model of the object being inspected to extract the desired features from the range image with the necessary accuracy and precision ?

Throughout our research we realised that some of our questions could not be answered in the way they were formulated, as they did not take into account the uncertainty

inherent in the inspection process. Thus, the first step of our investigation was to rephrase some of our original questions in order to make them more precise and closer to reality.

The main objective of an inspection process is to judge the quality of the part being inspected using the results of a measurement or a set of measurements. As measurements are never free of errors, inspection processes cannot be perfect and the inspection might eventually produce wrong judgements. The quality of a given inspection process is directly related to the probability of mistakes, or, in other words, to the failure rate of the inspection process.

The first two of our questions only make sense if we take into account the desirable failure rate of the inspection process, as the kinds of inspection we can do and the accuracy that we can achieve are directly related to the failure rate considered. Our approach was to try to establish a formal relation between the failure rates in the inspection of a manufactured object using a given tolerance zone and the accuracy and precision of the range sensor used.

Therefore, we concentrated on the evaluation of the failure rates of all the stages of the inspection process that we measured using two probabilities:

- the probability of not detecting that something is wrong (related to the sensitivity of the process) and
- the probability of saying that something is wrong when it is not (related to the reliability of the process).

Our attempt to answer the first two questions is expressed in the form of relations between the kind of range sensor used (defined by the pixel size and standard deviation of range measurement), the size of tolerance zone being inspected and the reliability and sensitivity of the inspection process.

In addressing the third question, the modelling of objects for inspection purposes was discussed in Chapter 3. In this chapter we analysed in detail the modelling issue and proposed a particular implementation of EDT models for inspection.

The fourth question is one of the main subjects of Chapter 4. Clearly, it is not possible to devise a general strategy to design a feature extraction process with a desired accuracy and precision for all possible kinds of features. Furthermore, it is not possible to design an inspection procedure which satisfies all the desired properties that we enumerated in Section 4.3.

Therefore, we concentrated our efforts on a general strategy for modelling the errors in the feature extraction process. Also, we tried to exemplify the use of the model shape for the extraction of many different kinds of features (*e.g.* roof edge points, step edge points, vertices and angles between planes and lines).

In general, from our experiments in Chapters 4 to 6, we can say that the inspection of mechanical parts using range data is viable in many cases such as straightness inspection, planarity inspection, angularity inspection and the measurement of waviness errors (as long as the correct sensor is used). However, there are some cases in which the use of dense range data is not ideal such as in the inspection of small holes.

From all that it was explored, we believe that we achieved our original objective of investigating the main issues related to the inspection of mechanical parts using dense range data.

It is important to observe, however, that we were not able to produce exact estimates of the failure rates of the inspection procedures developed, but only approximations of these values, and there is still a great deal of further research to be done for the determination of better approximations of the failure rates of inspection procedures.

In the next sections we present our final comments concerning the results obtained from Chapter 4 to 6. Possible directions for further research are also discussed.

7.2 Inspection of common manufactured objects

As a final comment on the results of Chapter 4 it is important to observe that through the formal definition of the inspection diagnosis and the implementation of the inspection procedures we were able to create a procedural definition for the semantics of the EDT models we described in Chapter 3, *i.e.*:

- The definitions of the I_p^{shape}, I_p^{size} and I_p^{pose} diagnoses in Chapter 4 are the procedural definition of the semantics of the DT nodes corresponding to the **ED-form**, **ED-size** and **ED-pose** types of tolerance defined in Chapter 3 according to Requicha's ideas.

- The implementation of the inspection procedures for straightness, planarity, segment size and hole size inspection defines a procedural semantic for the types of DT nodes described in Tables 3.1 and 3.2 according to the standards of tolerancing.

Another important comment about the results of our work is the fact that we were not able to find the exact failure rates associated with the inspection procedures we developed. This happens because of two main reasons:

- The failure rates were estimated using synthetic images that are just approximations of the actual range images.

- The theory developed to find the failure rates produces only estimates of the upper limits of the failure rates.

As a consequence of these facts the relations we found between the kind of range sensor used, the size of the tolerance zones considered and the reliability and sensitivity of the inspection processes we developed are only approximations.

There is a great amount of further research that can be done to improve the results obtained:

1. **Feature extraction**:

   The feature extraction strategy proposed in Chapter 4 can be improved in many ways:

   (a) **Statistical modelling of errors**: Improvement in the accuracy of the generation of the synthetic images to make the statistical modelling more accurate. This could be done by the realisation of experiments to determine

an accurate model of the errors in the range measurement and the determination of the behaviour of the sensor near the C_0 and C_1 discontinuities of the surface being imaged - see Appendix B.

(b) **Algorithms**: Research for the development of many different algorithms for feature extraction and construction of a library of algorithms.

(c) **Combination of algorithms**: Investigation of the use of techniques such as Kalman Filtering to combine different algorithms for extracting the same kind of feature. The objective would be to produce more precise and accurate algorithms.

2. **Reliability and sensitivity modelling**: Development of a new theory relating the uncertainty regions and the size of the tolerance zones being inspected to obtain a more accurate estimate of reliability and sensitivity. The main idea behind the research would be to build a theory that took into account the shape of the uncertainty regions and tolerance zones to obtain better estimates of reliability and sensitivity.

3. **Inspection diagnosis**:

   (a) **Test tolerance zone positioning**: Investigation on the use of optimisation techniques that do not use derivatives, such as the Rosenbrook and the Nelder and Mead methods (see [Bazaraa *et al.* 93]), for positioning the test tolerance zones during shape and size inspection.

   (b) **Pose inspection diagnosis I**: Analysis of alternative definitions for the pose inspection diagnosis in order to obtain a less demanding and more suitable definition.

   (c) **Pose inspection diagnosis II**: Research towards the combination of different estimates of pose to improve the accuracy in the determination of the datum pose.

4. **Implementation of inspection procedures**: Implementation of inspection procedures to inspect other kinds of features such as curves created by the intersection of planes and cones or cylinders.

In Chapter 4 we observed that an important phase in the inspection design was the determination of a global inspection plan to control the whole inspection process. The ideas presented in Chapter 4 could be used to develop such a planner using the following strategy:

1. **Determination of viewing conditions**: Use of simulations to determine the viewing conditions necessary to implement the inspection procedure associated with each DT node in the EDT model. This procedure would associate each of the DT nodes with a set of possible viewing points that could be used to check the DT node with the desired sensibility and reliability.

2. **Plan elaboration**: Processing all of the viewing conditions determined previously to determine the final inspection plan. The plan should make it possible to satisfy all the viewing restrictions using the smallest number of views of the part being inspected.

7.3 Registration of sculptured surfaces

As we can see from the results of Chapter 5 the multiscale approach adopted in the implementation of the modified ICP worked well and we were able to obtain the same accuracy as the original algorithm in a more efficient way.

Although the consistency checks added to the algorithm were good enough to eliminate matches that were clearly wrong, they were not able to eliminate all the wrong correspondences. This happened because the coarse sampling of data points made the consistency checks not very demanding which reduced their efficiency.

Despite being more efficient than the conventional ICP, the modified ICP still has a few characteristics related to its accuracy, efficiency (speed) and convergence properties that are less than ideal for inspection purposes. Further work towards the improvement of these characteristics could be done in the following aspects:

1. **Accuracy I**: Analysis of the possibility of implementing a variation of Horn's algorithm that takes into account the anisotropic characteristic of the sensor errors to improve the registration accuracy.

2. **Accuracy II**: Investigation of alternative methods to determine the closest point belonging to a B-spline model shape to a given 3D data point. The objective would be to improve the accuracy and efficiency of the fine approximation phase of the modified ICP algorithm of Section 4.4.

3. **Efficiency**: Implementation of the **C(P,X)** operator using the calculation off-line of Voronoi diagrams (as suggested in [Bentley *et al.* 80]) to reduce the computational time during the coarse approximation phase described in the modified ICP algorithm of Section 4.4.

4. **Convergence**: Investigation of alternative ways to determine a suitable initial estimate of pose to be used with the ICP.

5. **Consistency checks**: Development of different kinds of consistency checks and investigation on the influence of the checks on the convergence of the algorithm.

7.4 Waviness measurement

The use of the DFT for the detection and measurement of waviness deformations proved to be very effective. As we discussed in Chapter 6 our method presented a good sensitivity and was also able to produce good estimates of the amplitude of the waviness deformations and of their localisation within the range image.

Despite its qualities the method proposed is not robust to errors in the registration phase between model and data. Furthermore, as we observed in our experiments, modelling errors can also lead to wrong results of the deformation amplitude. This means that the method can only be used together with accurate models of the surfaces being inspected.

Further research could be done in:

1. **Modelling**: More accurate modelling of the typical periodic deformations of the grinding and milling operations to allow a more accurate determination of the sensitivity of the method.

2. **Efficiency**: Development of a parallel version of the algorithm using FFT algorithms.

3. **Implementation**: construction of specific hardware for the implementation of the algorithm using DSP chip processors. The main objective being the use on-line of the method to control the waviness errors during the manufacturing process.

4. **Generalisation**: Investigation of the viability of extending the method for use with other kinds of deformations.

7.5 Original contributions

In this thesis we investigated the use of dense range data for the inspection of machined parts. We concentrated our research on three issues that we judged to be the most relevant in our framework of research: modelling, registration and inspection.

The contributions of the thesis in these three issues are:

- **Modelling**
  - We presented a particular implementation of the EDT models that we extended to accommodate the representation of tolerances according to Requicha's theory.
- **Registration**
  - We discussed the use of the ICP algorithm for inspection purposes. The limits in the accuracy of the ICP algorithm as a function of the sensor errors and number of correspondences were determined.
  - A modified version of the ICP algorithm that uses a multiscale approach to make the algorithm more efficient and consistency checks to make it more robust to outliers was presented.
- **Inspection**

- We proposed a novel approach to the inspection of common manufactured objects using three distinctive stages: feature extraction, tolerance zone positioning and inspection diagnosis.
- A theoretical model of the reliability and sensitivity of the inspection procedures based on the statistical modelling of the errors in the feature extraction phase was developed. The method makes it possible to evaluate numerically the inspection procedures and uses Monte Carlo simulations to model the errors of the feature extraction phase.
- We proposed a procedural semantics to the EDT models through the formal definition of the inspection diagnosis. Examples using some of the more common tolerances described in the tolerancing standards were presented.
- A novel method for the detection, reconstruction and measurement of waviness errors based on the use of DFT was proposed and a model of the sensitivity and reliability of the method was developed.

Bibliography

[Aggarwal & Chien 88] J.K. Aggarwal and C.H. Chien. 3D structures from 2D images. In J. L. C. Sanz, editor, *Advances in Machine Vision.* Springer-Verlag, New York, 1988.

[Ahola *et al.* 85] R. Ahola, T. Heikkinen, and M. Manninen. 3D image acquisition by scanning time of flight measurement. In *Proc. Intl. Conf. on Advances in Image Processing and Pattern Recognition.* Pisa, Italy, December 10-12 1985.

[Ahuja & Abbot 93] N. Ahuja and A. L. Abbot. Active stereo - integrating disparity, vergence, focus, aperture and calibration for surface estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10), 1993.

[Aldelfeld 86] B. Aldelfeld. Rule-based approach to variational geometry. In *Knowledge Engineering and Computer Modelling in CAD*, pages 59–66. Butterworths, UK, 1986.

[Arun *et al.* 87] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3D point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5), September 1987.

[Asada *et al.* 86] M. Asada, H. Ichikawa, and S. Tsuji. Determining surface property by projecting a stripe patter. In *Proc. International Conference on Pattern Recognition.* Paris, France, IEEE-CS, IAPR, pp. 1439-1442, 1986.

[Ayache *et al.* 89] N. Ayache, J. D. Boissonat, E. Brunet, L. Cohen, J. P. Chieze, B. Geiger, O. Monga, J. M. Rocchisani, and P. Sander. *Building highly structured volume representations in 3D medical images.* In Computer Aided Radiology, Berlin, West Germany, 1989.

[Baker & Binford 80] H. H. Baker and T. O. Binford. Depth from edge intensity based stereo. In *Proc. 7th IJCAI.* Vancouver, British Columbia, Canada, IJCAI, pp 24-28, August 1980.

[Baker 81] H. H. Baker. Edge based stereo correlation. In *Proc. Image Understanding Workshop.* Colleg Park, MD., Sci-

ence Applications, Inc., McLean, Va., pp 168-175, April 1981.

[Ballard & Brown 82] D. H. Ballard and C. M. Brown. *Computer Vision.* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982.

[Banic *et al.* 87] J. Banic, S. Sizgoric, and R. O'Neill. Airbone scanning lidar bathymeter measures water depth. *Laser focus/Electro-optics*, pages 48–52, February 1987.

[Barnard 86] S. T Barnard. A stochastic approach to stereo vision. In *Proc. National Conference on AI.* AAAI-86, Philadelphia, Kaufmann, Los Altos, Calif., pp 676-680, August 11-15 1986.

[Bartlett *et al.* 88] S. L. Bartlett, P. J. Besl, C. L. Cole, R. Jain, D. Mukherjee, and K. D. S. Kifstad. Automatic sold joint inspection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(1), January 1988.

[Baumgart 72] B. G. Baumgart. Winged edge polyhedron representation. Research report stan-cs-320, Stanford, AI Laboratory, October 1972.

[Bazaraa *et al.* 93] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear programming - Theory and Algorithms.* John Wiley and Sons, 1993.

[Beheim & K. 86] G. Beheim and Fritsch K. Range finding using frequency-modulated laser diode. *Appl. Opt.*, 25(9):1439–1442, May 1986.

[Bentley 75] J. L. Bentley. Multidimensional search trees used for associative searching. *Comm. ACM*, 18(9), 1975.

[Bentley *et al.* 80] J. L. Bentley, B. W. Weide, and A. C. Yao. Optimal expected-time algorithms for closest point problems. *ACM Transactions on Mathematical Software*, 6(4), December 1980.

[Bernstein & Priess 89] N. S. Bernstein and K. Priess. Representation of tolerance information in solid models. In B. Ravani, editor, *Advances in Design Automaion - Vol I*, pages 37–48. Computer-Aided and Computational Design ASME, USA, 1989.

[Besl & McKay 92] P.J. Besl and N. McKay. A method for registration of 3D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[Besl 88a] P. J. Besl. Active optical range imaging sensors. In J. L. C. Sanz, editor, *Advances in Machine Vision.* Springer-Verlag, New York, 1988.

[Besl 88b] P. J. Besl. Geometric modelling and computer vision. Research report gmr-6248, General Motors Research Laboratories, April 1988.

[Bickel *et al.* 85] G. Bickel, G. Hausler, and M. Maul. Triangulation with expanded range of depth. *Opt. Engrg.*, 24(6):975–979, 1985.

[Binger & Harris 87] N. Binger and S. J. Harris. Applications of laser radar technology. *Sensors*, 4(4):42–44, 1987.

[Bispo & Fisher 94a] E. M. Bispo and R. Fisher. Free-form surface matching for surface inspection. In *IMA Mathematics of Surfaces VI Conference.* London, September 1994.

[Bispo & Fisher 94b] E. M. Bispo and R. Fisher. Inspection of free-form surfaces using dense range data. In *European Symposium on Optics for Productivity in Manufacturing - Automated Visual Inspection.* Frankfurt, Junho 1994.

[Boehnlein & Harding 86] A. J. Boehnlein and K. G. Harding. Adaptation of a parallel architecture computer to phase-shifted moire interferometry. In *Proc. SPIE Conference on Optics, Illumination, and Image Sensing for Machine Vision*, volume 728. Cambridge, Mass., pp. 235-242, October 30-31 1986.

[Boerner & Strecker 88] H. Boerner and H. Strecker. Automatic x-ray inspection of aluminum castings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(1), January 1988.

[Boyer & Kak 87] K. L. Boyer and A. C. Kak. Color encoded structured light for rapid active ranging. *IEEE Transactions Robotics and Automation*, 9(1):14–21, January 1987.

[Boyer *et al.* 86] K. L. Boyer, A. J. Vayda, and A. C. Kak. Robotic manipulation experiments using structural stereopsis for 3D vision. *IEEE Expert*, 1(3):73–94, Fall 1986.

[Brooks 79] M. J. Brooks. Surface normals from closed paths. In *Proc. 6th IJCAI.* Tokyo, Japan, pp. 98-101, August 20-23 1979.

[Bruss 81] A. Bruss. *Shape from shading and bounding contour.* Unpublished PhD thesis, Department of Electrical Engineering and Computer Science, Cambridge, Mass., 1981.

[BS308 72a] BS308. *Engineering drawing practice, Part 2: Dimensioning and tolerancing of size.* British Standards Institution, UK, 1972.

[BS308 72b] BS308. *Engineering drawing practice, Part 3: Geometrical tolerancing.* British Standards Institution, UK, 1972.

[Byungil & Burger 91] K. Byungil and P. Burger. Depth and shape from shading using the photometric stereo method. *CVGIP : Image understanding*, 54(3), 1991.

[Cardenasgarcia *et al.* 94] J. F. Cardenasgarcia, S. Zheng, and F. Z. Shen. Implementation and use of an automated projection Moire experimental setup. *Optics and Lasers in Engineering*, 21(1):77–98, 1994.

[Cardew-hall *et al.* 93] M. J. Cardew-hall, T. Labans, G. West, and P. Dench. A method of representing dimensions and tolerances on solid based freeform surfaces. *Robotics and Computer-Integrated Manufacturing*, 10(3), 1993.

[Chambers & Brown 86] R. V. Chambers and R. E. Brown. Coordinate measurement of bevel gear teeth. Paper sae 871645, SAE Tech., 1986.

[Chen & Medioni 92] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Imagine and Vision Computing*, 10(3), April 1992.

[Chen *et al.* 92] L. L. Chen, L. S. Shew, and B. G. Sun. Measurement of length with high accuracy and large range. *Optics and Laser*, 17(1):27–33, 1992.

[Cheng & Menq 95] W. L. Cheng and C. H. Menq. Integrated laser CMM system for the dimensional inspection of objects made of soft material. *Intl. Journal of Advanced Manufacturing Technology*, 10(1), 1995.

[Cline *et al.* 84] H. E. Cline, W. E. Lorenson, and A. S. Holik. Automated moire contourning. *Appl. Opt.*, 23(10):1454–1459, 1984.

[Cochran & Medioni 92] S. D. Cochran and G. Medioni. 3D surface description from binocular stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10), 1992.

[Conners *et al.* 83] R. W. Conners, C. W. McMillin, and R. E. Lin, K. Vasquez-Espinosa. Identifying and locating surface defects in wood: Part of an automated lumber processing system. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(6), 1983.

[Corp. 87] Technical Arts Corp. *Product Literature.* Redmond, Wash, 1987.

[Corporation 87] Cyberoptics Corporation. *Production information.* Minneapolis, Minn., 1987.

[Deboor 78] C. Deboor. *A practical guide to splines.* Springer-Verlag, New York, 1978.

[Diffracto. 87] Diffracto. *Laser probe digital ranging sensor.* Product Literature, Diffracto, Ltd.,Windsor, Otario, Canada, 1987.

[Dinauer *et al.* 94] W. R. Dinauer, N. A. Duffle, and M. L. Philpott. Error compensation algorithms for sculptured surface production. *Transactions of the ASME Journal of Engineering for Industry*, 116, May 1994.

[Duncan *et al.* 80] J. P. Duncan, D. P. Dean, and G. C Pate. Moiré contougraphy and computer aided replication of human anatomy. *Engrg. in Medicine*, 9(1), 1980.

[Engelhardt 91] K. Engelhardt. Acquisition of 3D data by focus sensing utilizing the Moire effect on CCD cameras. *Applied Optics*, 30(11):1401–1407, 1991.

[Etesami 88] F. Etesami. Tolerance verification through manufacturing part modelling. *Journal of Manufacturing Systems*, 7(3):223–232, 1988.

[Faugeras & Herbert 86] O. D. Faugeras and M. Herbert. The representation, recognition and location of 3D objects. *Int. J. Robotic Res.*, 5(3), Fall 1986.

[Fisher 89] R. B. Fisher. *From surface to objects: recognising objects using surface information and object models.* John Wiley and Sons, 1989.

[Fitzgerald 81] W. J. Fitzgerald. Using line drawings to determine the proportions of line drawings in computer graphics. *Computer-Aided Design*, 13(6), 1981.

[Font *et al.* 92] B. V. Font, M. S. Drew, and M. Brockington. Recovering shade from color images. *Lecture Notes in Computer Science*, 588, 1992.

[Fujihara *et al.* 94] K. Fujihara, J. Serizawa, and Y. Furukawa. A pattern inspection system for high-density printed wiring boards. *Fujitsu Scientific and Technical Journal*, 39(1), 1994.

[Gao *et al.* 93] J. X. Gao, K. Case, and N. Gindy. Geometric elements for tolerance definition in feature-based product models. *Intl. Journal od Computer Integrated Manufacturing*, 25(9):1439–1442, 1993.

[Gopin & Gossard 79] A. M. Gopin and D. C. Gossard. Symbolic dimensioning. In *Proceedings 1st Ann. Conf. Computer Graphics in CAD/CAM.* Massachusetts Institute of Technology, USA, April 1979.

[Gossard *et al.* 88] D. C. Gossard, R. P. Zuffante, and H. Sakurai. Representing dimensions, tolerances, and features in mcae systems. *IEEE Comput. Graph. and Applic.*, 8(2):51–59, 1988.

[Gottwald & Berner 87] R. Gottwald and W. Berner. *The knew kern system for positioning and automated coordinate evaluation; automated 3D coordinate determination.* Product Literature, Kern Instruments, Brewester, N. Y., and Aarau, Switzerland, 1987.

[Grimson & Lozano-Perez 84] W. E. L. Grimson and T. Lozano-Perez. Model-based recognition and localization from sparse range or tactile data. *Int. J. Robotic Res.*, 3(3), 1984.

[Grimson 81] W. E. L. Grimson. *From images to surfaces.* MIT Press, Cambridge, Mass., 1981.

[Grimson 84] W. E. L. Grimson. Computational experiments with a feture based stereo algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(1), 1984.

[Gu *et al.* 94] R. W. Gu, T. Yoshizawa, and Y. Utani. One step phase shift 3D surface profilometry with grating projection. *Optics and Laser Engineering*, 21(1):61–75, 1994.

[Gueziec & Ayache 94] A. Gueziec and N. Ayache. Smoothing and matching of 3d space curves. *International Journal of Computer Vision*, 12(1), 1994.

[Haggren & Leikas 87] H. Haggren and E. Leikas. Map-vision : The photogrammetric machine vision system. In *Proc. Vision 87 Conference.* Detroit, Mich., June 8-11 1987.

[Halioua & Srinivasan 87] M. Halioua and V. Srinivasan. *Method and apparatus for surface profilometry.* New York Institute of Technology, Old Westbury, N. Y. U. S. Patent 4,641,972, 1987.

[Hall *et al.* 87] E. L. Hall, J. B. K. Tio, C. A. McPherson, and F. A. Sadjadi. Measuring curved surfaces for robot vision. *Computer*, 15(12):42–54, December 1987.

[Hamilton 96] S. Hamilton. *3D Scanners.* Product Literature, London, United Kingdom, 1996.

[Hara *et al.* 83] Y. Hara, N. Akiyama, and K. Karakasi. Automatic inspection system for printed circuit boards. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(6), November 1983.

[Hara *et al.* 95] Y. Hara, H. Doi, K. Karasaki, T. Iida, T. Furutani, S. Kitamura, N. Minatani, and S. Shinada. Automated inspection of printed-circuit board patterns referenced to CAD data. *Intl. Journal of the Japan Society for Precision Engineering*, 29(4), 1995.

[Harding & Tait 86] K. G. Harding and R. Tait. Hybrid high accuracy structured light. In *Proc. SPIE Conference on Optics, Illumination, and Image Sensing for Machine Vision*, volume 728. Cambridge, Mass., pp. 132-145, October 30-31 1986.

[Heikkinen *et al.* 86] T. Heikkinen, R. Ahola, M. Maninnen, and M. Myllyla. Recent results of the performance analysis of a 3D sensor based on time of flight. In *Proc. SPIE Quebec Intl. Symp. on Optical and Optoelectronic Applied Sci. and Engrg.*, 1986.

[Hendengren 88] K. H. Hendengren. Methodology for automatic image-based inspection of industrial objects. In J. L. C. Sanz, editor, *Advances in Machine Vision*. Springer-Verlag, New York, 1988.

[Hersman *et al.* 87] M. Hersman, F. Goodwin, S. Kenyon, and A. Slotwinski. Coherent laser radar application to 3D vision and metrology. In *Proc. Vision'87 Conf.* Detroit, Mich., June 8-11 1987.

[Hild *et al.* 95] R. Hild, G. Nitzsche, and U. Altenburger. The challenges of color for wafer inspection from the viewpoint of partially coherent imaging theory. *Microeletronic Engineering*, 26(3), 1995.

[Hillyard & Braid 78] A. C. Hillyard and I. C. Braid. Characterizing non-ideal shapes in terms of dimensions and tolerances. *Comput. Graph.*, 12(3):234–238, 1978.

[Hillyard & Braid 82] A. C. Hillyard and I. C. Braid. Analysis of dimensions and tolerances in computer-aided design. *Comput.-Aided Des.*, 14(4):209–214, 1982.

[Hillyard 78] R. C. Hillyard. *Dimensions and tolerances in shape design.* Unpublished PhD thesis, Computer Laboratory, University of Cambridge, 1978.

[Hiroi *et al.* 93] T. Hiroi, K. Yoshimura, T. Ninomiya, T. Hamada, Y. Nakagawa, S. Mio, K. Karasaki, and H. Sasaki.

Solder joint inspection using air stimulation speckle vibration detection method and fluorescence detection method. *IEICE Transactions on Information and Systems*, E76D(10), 1993.

[Hoffman 82] P. Hoffman. Analysis of tolerances and process inaccuracies in discret part manufacturing. *Comput.-Aided Des.*, 14(2):3–88, 1982.

[Horn & Brooks 90] B. K. P. Horn and M. J. Brooks. Height and gradient from shading. *Internationa Journal of Computer Vision*, 5(1), 1990.

[Horn & Harrys 87] B. K. Horn and J. G. Harrys. Closed-form solution of absolute orientation using unit quaternions. *J. Optical Society of America*, 4(4), 1987.

[Horn 75] B. K. P. Horn. Obtaining shape from shading information. In P. H. Winston, editor, *The psychology of computer vision*, pages 115–155. McGraaw-Hill, New York, 1975.

[Horn 77] B. K. P. Horn. Understand image intensities. *Artificial Intelligence*, 8(2):201–231, April 1977.

[Hymarc 87] Hymarc. *Product information.* Ottawa, Ontario, Canada, 1987.

[Idesawa & Yatagai 84] M. Idesawa and Y. Yatagai. 3D shape input and processing by moire technique. In *Proc. 5th International Conference on Pattern Recognition.* Miami, Florida, IEEE-CS, New York, pp. 1085-1090, December 1984.

[Idesawa *et al.* 77] M. Idesawa, Y. Yatagai, and T. Soma. Scanning moire method and automatic measurement of 3D shapes. *Appl. Opt.*, 16(8):2152–2162, 1977.

[Inokuchi *et al.* 84] S. Inokuchi, K. Sato, and F. Matsuda. Range imaging system for 3D object recognition. In *Proc. International Conference on Pattern Recognition.* Montreal, Canada, pp. 806-808, July 30 - August 2 1984.

[Iwahory *et al.* 94a] Y. Iwahory, H. Tanaka, R. G. Woodham, and N. Ishii. Photometric stereo for specular surface shape-based on neural network. *IEICE - Transactions on information and systems*, E77D(4), 1994.

[Iwahory *et al.* 94b] Y. Iwahory, R. G. Woodham, H. Tanaka, and N. Ishii. Moving point light source photometric stereo. *IEICE - Transactions on information and systems*, E77D(8), 1994.

[Jackins & Tanimoto 80] C. L. Jackins and S. L. Tanimoto. Oct-trees and their use in representing three-dimensional objects. *CVGIP : Image understanding*, 14(3), 1980.

[Jain *et al.* 86] R. Jain, R. Rao, A. Kayalp, and C. Cole. Machine vision for semicondutor wafer inspection. In H. Freeman, editor, *Machine Vision for Inspection and Mesurement.* Academic Press, Inc. Harcout Brace Jovanich, 1986.

[Jalkio *et al.* 85] J. Jalkio, R. Kim, and S. Case. Triangulation based range sensor design. *Opt. Engrg.*, 24(6):966–974, November-December 1985.

[Johnson 85] R. H. Johnson. Dimensioning and tolerancing final report. Techinical report r-84-gm-02.2, CAM-I, USA, May 1985.

[Jones & Taylor 94] A. G. Jones and C. J. Taylor. Robust shape from shading. *Image and vision computing*, 12(7), 1994.

[Juha & Donahue 87] M. Juha and J. Donahue. *Improving automated SMT inspection with 3D Vision.* Product Literture, Photonic Automation, Inc, Santa Ana, Calif., 1987.

[Juster 88] N. P. Juster. *A graph based approach to tolerance analysis.* Unpublished PhD thesis, Dep. Mechanical Engineering, University of Leeds, UK, 1988.

[Juster 92] N. P. Juster. Modelling and representation of dimensions and tolerances: a survey. *Computer Aided Design*, 24(1), January 1992.

[Kanade & Fuhrman 87] T. Kanade and M. Fuhrman. A noncontact optical prosimity sensor for measuring surface shape. In T. Kanade, editor, *Three-dimensional Machine Vision*, pages 151–194. Kluwer Academic, Boston, 1987.

[Kanade & Okutomi 94] T. Kanade and M. Okutomi. A stereo matching algorithm with an adaptative window - theory and experiment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9), 1994.

[Khalaj *et al.* 94] B. H. Khalaj, H. K. Aghajan, and T. Kailath. Patterned wafer inspection by high-resolution spectral estimation techniques. *Machine Vision and Applications*, 7(3), 1994.

[Kim & Aggarwal 85] Y. C. Kim and J. K. Aggarwal. Finding range from stereo images. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition.* San Francisco, California, 19-23, IEEE, New York, June 1985.

[Kimura *et al.* 86] E. Kimura, H. Suzuki, and L. Wingard. A uniform approach to dimensioning and tolerancing in product modelling. *Comput. Aided Produt. Eng.*, 1986.

[Kinoshita *et al.* 86] G. Kinoshita, M. Idesawa, and S. Naomi. Robotics range sensor with projections of a bright ring. *J. Robotic Systems*, 3(3):249–257, March 1986.

[Kishon *et al.* 90] E. Kishon, T. Hastie, and H. Wolfson. 3D curve matching using splines. In *Proc. 1st European Conference in Computer Vision.* Antibes, France, April 1990.

[Klages & Wilson 94] E. J. Klages and R. M. Wilson. Apm - an automated inspection programming system. *Intl. Journal of Computer Integrated Manufacturing*, 7(6), 1994.

[Komorowski *et al.* 96] J. P. Komorowski, S. Krishnakumar, R. W. Gould, N. C. Bellinger, F. Karpala, and O. L. Hageniers. Double-pass retroreflection for corrosion detection in aircraft structures. *Material Evaluation*, 54(1), 1996.

[Komorya & Tani 91] K. Komorya and K. Tani. Development of a laser range sensor for a mobile robot and its landmark detection performance. *Journal of Mechanical Engineering Laboratory*, 45(4), 1991.

[Kontsevich *et al.* 94] L. L. Kontsevich, A. P. Petrov, and I. S. Vergelskaya. Reconstruction of shape from shading in color images. *Journal of the Optical Society of America - Optics, Image Science and Vision*, 11(3), 1994.

[Kosakai *et al.* 93] I. Kosakai, T. Tada, M. Hirata, H. Kamei, K. Aburatani, and H. Nakata. Surface inspection system for billets, bars and wire rods. *Tetsu to Hagane-Journal of the Iron and Steel Institute of Japan*, 79(7), 1993.

[Labuz & McVey 86] J. Labuz and E. S. McVey. Camera and projector motion for range mapping. In *Proc. SPIE Conference on Optics, Illumination and Image Sensing for Machine Vision*, volume 728. Cambridge, Mass., October 30-31 1986.

[Landman & Robertson 86] M. M. Landman and S. G Robertson. A flexible industrial system for automated 3D inspection. In *Proc. SPIE Conference on Optics, Illumination and Image Sensing for Machine Vision*, volume 728. Cambridge, Mass., pp. 676-680, October 30-31 1986.

[Lee & Kuo 93] K. M. Lee and C. J. Kuo. Surface reconstruction from photometric stereo images. *Journal of the Optical Society of America Optics-Image Science and Vision*, 10(5), 1993.

[Lee & Kuo 94] K. M. Lee and C. J. Kuo. Shape from shading with perspective projection. *CVGIP: Image understanding*, 59(2), 1994.

[Leseberg 92] D. Leseberg. Small-distance position sensing by using holograms of conical waves. *Optic Letters*, 17(13), 1992.

[Lewis & Johnston 77] R. A. Lewis and A. R. Johnston. A scanning laser range finder for a robotic vehicle. In *Proc. 5th Intl. Joint Conference on Artificial Intelligence.* Cambridge, Mass., August 22-25 1977.

[Light 80] R. A. Light. *Symbolic dimensioning in computer aided design.* Master's Thesis Massacussetts Institute of Technology, USA, 1980.

[Lin *et al.* 88] V. C. Lin, D. C. Gossard, and R. A. Light. A variational geoemtry in computer-aided design. *Computer Graph.*, 15(3):171–177, 1988.

[Lorenz 84] R. D. Lorenz. *Theory and design of optical/electronic probes for high performance measurement of parts.* Unpublished PhD thesis, University of Wisconsin-Madison, Madison, Wisc., 1984.

[Lorusso & Eggert 95] A. Lorusso and D. W. Eggert. A comparison of four algorithms for estimating 3-D rigid transformations. In *British Machine Vision Conference.* University of Birmingham, Birmingham, September 1995.

[Marapanese & Trivedi 94] S. B. Marapanese and M. M. Trivedi. Multi-primitive hierarchical (MPH) stereo analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(3), 1994.

[Marr & Poggio 79] D. Marr and T. Poggio. A theory of human stereo vision. In *Proc. Roy. Soc. London, B207.* pp 301-328, October 1979.

[Marr & Poggio 81] D. Marr and T. Poggio. A computer implementation of a theory of human stereo vision. In *Proc. Roy. Soc. London, B292.* pp 217-253, 1981.

[Marr 82] D. Marr. *Vision.* W. H. Freeman and Company, New York, 1982.

[Marshall 89] A. D. Marshall. *The automatic inspection of machined parts using three dimensional range data and model based matching techniques.* Unpublished PhD thesis, University of Wales, College of Cardiff, UK, April 1989.

[Marshall *et al.* 91] A. D. Marshall, R. R. Martin, and D. Hutber. Automatic inspection of mechanical parts using geometric models and laser range finder data. *Image and Vision Computing*, 9(6), 1991.

[Matsumoto *et al.* 92] T. Matsumoto, Y. Kitagawa, M. Adachi, and A. Hayashi. Moire topography for 3D dimensional profile mesurement using the interference fringes of a laser. *Optical Engineering*, 31(12):2668–2673, 1992.

[Medland *et al.* 93] A. J. Medland, G. Mullineux, and A. H. Rentoul. Measurement of features by a sample-grid approach. *Computer Integrated Manufacturing Systems*, 33(5):725–740, 1993.

[Menq & Yau 92] C. Menq and H. Yau. Automated precison measurement of surface profile in CAD-directed inspection. *IEEE Transactions on Robotics and Automation*, 8(2):268–278, April 1992.

[Menq *et al.* 92] C. Menq, H. Yau, and C. Wong. An intelligent planning environment for automated dimensional inspection using coordinate measuring machines. *Transactions of the ASME Journal of Engineering for Industry*, 114:222–230, May 1992.

[Michael *et al.* 91] R. M. Michael, A. D. Jepson, and J. K. Tsotos. Techniques for disparity measurement. *CVGIP : Image understanding*, 53(1), 1991.

[Miller & Wagner 87] G. L. Miller and E. R. Wagner. An optical rangefinder for autonomous robot cart navigation. In *Proc. SPIE Industrial Electronics.* Cambridge, Mass., pp. 676-680, November 1987.

[Mundy & Porter 87] J. L. Mundy and G. B. Porter. A three-dimensional sensor base on structured light. In T. Kanade, editor, *Three-dimensional Machine Vision*, pages 3–62. Kluwer Academic, Boston, 1987.

[Naidu *et al.* 90] D. K. Naidu, R. B. Fisher, and G. Cameron. *User guide for the laser range image acquisition package.* University of Edinburgh, Dept. of AI, Imagine software paper num. 18, 1990.

[Nakagawa & Ninomiya 87] Y. Nakagawa and T. Ninomiya. Three-dimensional vision systems using the structured light method for inspecting solder joints and assembly robots. In T. Kanade, editor, *Three-dimensional Machine Vision*, pages 543–565. Kluwer Academic, Boston, 1987.

[Nayar *et al.* 95] S. K. Nayar, M. Watanabe, and M. Noguchi. Real-time focus range sensor. In *Proc. Fifth Intl. Conf. on Computer Vision.* Cambridge, Massachusetts, June 20-23 1995.

[Nevatia & Binford 73] R. Nevatia and T. O. Binford. Structured description of complex objects. In *Proc. 3rd International Joint Conference on Artificial Intelligence.* Stanford, California, pp. 641-647, August 20-23 1973.

[Newman & Jain 95] T. S. Newman and A. K. Jain. A system for 3D CAD-based inspection using range images. *Pattern Recognition*, 28(10), 1995.

[Nitzan *et al.* 86] D. Nitzan, R. Bollees, J. Kremers, and P. Mulgaonkar. 3D vision for robot applications. In *Proceedings NATO Workshop on Knowledge Engineering for Robotic Applications.* Maratea, Italy, May 12-16 1986.

[Ozeki *et al.* 86] O. Ozeki, T. Nakano, and S. Yamamoto. Real-time range measurement device for 3D object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(4), 1986.

[Page 88] A. Page. Segmentation algorithms. Internal report no. aoi/tr/basr/880201, BAe, Sowerby Research Centre, 1988.

[Parden & Newell 84] G. Parden and R. G. Newell. A dimension based parametric design system. In *Proceedings CAD.* Butterwoths, UK, 1984.

[Pau 82] L. F. Pau. *Computer vision for electronics manufacturing.* Plenum Press, New York and London, 1982.

[Pelowski 86] K. R. Pelowski. 3D measurement with machine vision. In *Proc. Vision 86 Conference.* Detroit, Mich., June 8-11 1986.

[Pentland 84] A. P. Pentland. Local shading analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:170–187, March 1984.

[Petrov 93] A. P. Petrov. Obtaining shape from color shading. *Color Research and Application*, 18(6), 1993.

[Pierce *et al.* 94] B. L. Pierce, D. J. Shelton, H. G. Longbotham, S. Baddipudi, and P. Yan. Automated inspection of through hole solder joints utilizing x-ray imaging. *IEEE Aerospace and Electronic Systems Magazine*, 9(2), 1994.

[Piptone & Marshall 83] F. J. Piptone and T. G. Marshall. A wide-field scanning triangulation rangefinder for machine vision. *Int. J. Robotics Res.*, 2(1), Spring 1983.

[Ponce *et al.* 92] J. Ponce, A. Hoogs, and J. K. David. On using CAD models to compute the pose of curved 3d objects. *CVGIP : Image understanding*, 55(2), March 1992.

[Popplestone *et al.* 75] R. J. Popplestone, C. M. Brown, A. P. Ambler, and G. F. Crawford. Forming models of plane-and-cylinder faceted bodies from light stripes. In *Proc. 4th International Joint Conference on Artificial Intelligence.* Tbilisi, Georgia, USSR, pp. 664-668, September 1975.

[Potmesil 83] M. Potmesil. Generating models of solid objects by matching 3D surface segments. In *Proceedings 8th IJCAI.* Karlshure, Germany, pp 1089-1093, August 1983.

[Potsdamer & Altschuler 82] J. Potsdamer and M. Altschuler. Surface measurement by espace-encoded project beam system. *Computer Graphics Image Processing*, 18:1–17, 1982.

[Poussart & Laurendeau 88] D. Poussart and D. Laurendeau. 3D sensing for industrial computer vision. In J. L. C. Sanz, editor, *Advances in Machine Vision.* Springer-Verlag, New York, 1988.

[R1101 82] ISO R1101. *Technical drawings - tolerances of form and position.* International Standards Organization, Switzerland, 1982.

[Requicha 77] A. A. G. Requicha. Part and assembly description languages: dimensioning and tolerancing. Technical report tm-19, Production Automation Project, University of Rochester, USA, 1977.

[Requicha 80] A. A. G. Requicha. Representations for rigid solids: Theory, methods, and systems. *Computing Surveys*, 12(4), December 1980.

[Requicha 83] A. A. G. Requicha. Towards a theory of geometrial tolerancing. *Int. Journnal of Robot. Res.*, 2(4):45–60, 1983.

[Requicha 86] A. A. G. Requicha. Offseting operations in solid modelling. *Computer Aided Geoemtrical Design*, 3:129–148, 1986.

[Rioux & Blais 86] M. Rioux and F. Blais. Compact 3D camera for robotic applications. *J. Opt. Soc. Amer.*, 3(9):1518–1521, September 1986.

[Rioux 84] M. Rioux. Laser range finder based upon synchronized scanners. *Appl. Opt.*, 23(21):3837–3844, 1984.

[Rocker 74] F. Rocker. Localization and classification of 3D objects. In *Proceedings 3nd International Conference on Pattern Recognition.* Copenhagen, Denmark, pp. 527-528, August 1974.

[Rosenfeld & Tsikos 86] J. P. Rosenfeld and C. J. Tsikos. High-speed space encoding projector for 3D imaging. In *Proc. SPIE Conference on Optics, Illumination and Image Sensing for Machine Vision*, volume 728. Cambridge, Mass., pp. 146-151, October 30-31 1986.

[Ross 78] J. A. Ross. *Methods and systems for 3D measurement.* U.S. Patent 4, 199, 253, 1978.

[Roy & Liu 88] U. Roy and C. R. Liu. Feature based representational scheme of a solid modeler for providing dimensioning and tolerancing information. *Robotic and Computer Integr. Manuf.*, 4(3):335–345, 1988.

[Sahoo & Menq 91] K. C. Sahoo and C. Menq. Localization of 3-d objects having complex sculptured surfaces using tactile sensing and surface description. *Transactions of the ASME Journal of Engineering for Industry*, 113:85–93, February 1991.

[Sampson 87] R. E. Sampson. 3D range sensor via phase shift detection. (insert). *IEEE Computer*, 20(8):23–24, August 1987.

[Sanderson *et al.* 88] A Sanderson, L. E. Weiss, and S. K. Nayar. Automatic x-ray inspection of aluminum castings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(1), January 1988.

[Schewe & Forstner 86] H. Schewe and W. Forstner. The program palm for automatic line and surface measurement using image matching techniques. In *Proc. Symposium Intl. Soc. for Photogrammetry and Remote Sensing*, volume 26. Part3/2, From analog to digital, pp. 608-622, 1986.

[Schultz 94] H. Schultz. Retrieving shape information from multiple images of a specular surface. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2), 1994.

[Selcom. 87] Selcom. *Optocator product information.* Valdes, N. C., Partille, Sweden, Krefeld, West Germany, 1987.

[Servo-Robot 87] Servo-Robot. *Product information.* Boucherville, Quebec, Canada, 1987.

[Shirai 71] Y. Shirai. Recognition of polyhedra with a range finder. *Pattern Recognition*, 4:80–87, 1971.

[Silvaggi *et al.* 86] C. Silvaggi, F. Luk, and W. North. Positioning/dimension by structure light. *Exper. Tech.*, October 1986.

[Silven *et al.* 88] O. Silven, I. Virtanen, T. Westman, and T. Piironen. A design data-based visual inspection ystem for printed wiring. In J. L. C. Sanz, editor, *Advances in Machine Vision.* Springer-Verlag, New York, 1988.

[Smith 94] R. Smith. A quick eye. *Image Processing*, 6(1), 1994.

[Solid Photography 77] Inc. Solid Photography. *Optocator product information (now Robotic Vision Systems, Inc.).* Hauppage, N.Y., 1977.

[Srinivasan *et al.* 85] V. Srinivasan, H. C. Liu, and M. Halioua. Automated phase measuring profilometry: A phase-mapping approach. *Appl. Opt.*, 24(2):185–188, 1985.

[Stein 91] F. Stein. Structural hashing: Efficient 3-d object recognition. In *Proc. Int. Conf. Comput. Vis. Patt. Recog.* Hawaii, June 1991.

[Stein 92] F. J. Stein. *Structural indexing for object recognition.* Unpublished PhD thesis, IRIS, University of Southern California, April 1992.

[Sterki & Giinter 82] A. Sterki and A. Giinter. Cnc measuring center for small and medium-sized gears. *Industrial and Production Engineering*, 11(2), 1982.

[Strat 79] T. M. Strat. *A numerical method for shape from shading from a single image.* Master Thesis. Department of Electrical Engineering and Computer Science, Cambridge, Mass., 1979.

[Suresh *et al.* 83] B. R. Suresh, R. A. F. Kouski, T. S. Levitti, and J. E. Overlend. A real-time automated inspection for hot steel slabs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(6), November 1983.

[Suzuki *et al.* 85] H. Suzuki, F. Kimura, and T. Sata. Treatment of dimensions on product modelling concept. In *Design and Synthesis*, pages 491–496. North Holland, 1985.

[Svetkoff *et al.* 84] D. J. Svetkoff, P. F. Leonardo, R. E. Sampson, and R. C. Jain. Techniques for real-time feature extraction using range information. In *Proceedings SPIE Conference on Optics, Illuminations, and Image Sensing for Machine Vision*, volume 521. Vol. 728, Cambridge, Mass., pp. 216-226, October 30-31 1984.

[Szeliski 91] R. Szeliski. Fast shape from shading. *CVGIP : Image understanding*, 53(2), 1991.

[Takagi *et al.* 86] Y. Takagi, Y. Kojima, M. Yoshida, and H. Awane. A laser-applied three-dimensional profile measurement system. *Hitachi Review*, 35(1), 1986.

[Terzopoulos *et al.* 87] D. Terzopoulos, A. Witkin, and M. Kass. Stereo matching as constrained optimization using scale continuation methods. In *Proc. SPIE Conference on Optical and Digital Pattern Recognition.* Los Angeles, Calif., SPIE, Bellingham, Wash, January 13-15 1987.

[Todd 86] P. Todd. An algorithm for determining consistency and manufacturability of dimensioned drawings. In *Knowledge engineering and Computer Modelling in CAD*, pages 36–41. Butterworths, UK, 1986.

[Turner 87] J. U. Turner. *Tolerances in computer-aided design.* Unpublished PhD thesis, Rensselaer Polytechnique Institute, USA, 1987.

[Urquhart *et al.* 93] C. W. Urquhart, J. P. Siebert, J. P. McDonald, and R. J. Fryer. Active animate stereo vision. In *Proc. 4th British Machine Vision Conference.* University of Surrey, Guilford, pp. 55-65, September 21-23 1993.

[Vuylsteke & Oosterlinck 86] P. Vuylsteke and A. Oosterlinck. 3D perception with a single binary coded illumination pattern. In *Proceedings, Proc. SPIE Conference on Optics, Illumination and Image Sensing for Machine Vision.* Cambridge, Mass., pp. 195-202, October 30-31 1986.

[Wagner 86] J. F. Wagner. Sensors for dimensional measurement. In *SPIE Conference on Optics, Illumination and Image Sensing for Machine Vision*, volume 728. Cambridge, Mass. pp 173-182, October 1986.

[Walker *et al.* 91] M. W. Walker, L. Shao, and R. A. Volz. Estimating 3-D location parameters using dual number quaternions. *CVGIP: Image understanding*, 54(3), 1991.

[Wang & Aggarwal 86] Y. E. Wang and J. K. Aggarwal. Inferring local surface orientation with the aid of surface code. In *Proceedings IEEE Workshop on Computer Vision : Representation and Control*, volume 728. Cambridge, Mass., pp. 132-145, October 1986.

[Wei & Gini 83] D. Wei and M. Gini. The use of taper light beam for object recognition. In R. Pugh, editor, *Robot Vision*, pages 143–156. IFS Publications, U. K., Springer-Verlag, Berlin, Germany, 1983.

[Wilder 86] J. Wilder. Finding and evaluating defects in glasses. In H. Freeman, editor, *Machine Vision for Inspection and Mesurement.* Academic Press, Inc. Harcout Brace Jovanich, 1986.

[Will & Pennington 84] P. M. Will and K. S. Pennington. Grid coding : A novel technique for image processing. In *Proc. IEEE.* pp. 216-226, June 1984.

[Woodham 79] R. J. Woodham. A cooperative algorithm for determining surface orientation from a single view. In *Proc. 5th IJCAI.* Cambridge, Mass., pp. 635-641, August 22-25 1979.

[Y14.5M 82] ANSI Y14.5M. *Dimensioning and tolerancing.* American Society of Mechanical Engineers, USA, 1982.

[Yamamoto *et al.* 86] H. Yamamoto, K. Sato, and S. Inokuchi. Range imaging system based on binary image accumulation. In *Proc. International Conference on Pattern Recognition.* Paris, France,IEEE, New York, pp. 233-235, 1986.

[Yang & Menq 93] B. Yang and C. Menq. Compensation for form error of end-milled sculptured surfaces using discrete measurement data. *Internations Journal Mach. Tools Manufact.*, 33(5):725–740, 1993.

[Yau & Menq 92] H. Yau and C. Menq. An automated dimensional inspection environment for manufatured part using coordinate mesuring machines. *International Journal of Production Research*, 30(7):1517–1536, 1992.

[Yau & Menq 93] H. Yau and C. Menq. Concurrent process planning for finish milling and dimensional inspection of sculptured surfaces in die and mould manufacturing. *International Journal of Production Research*, 31(11):2709–2725, 1993.

[Yau & Menq 95] H. Yau and C. Menq. Automated CMM path planning for dimensional inspection of dies and molds having complex-surfaces. *Intl. Journal of Machine Tools and Manufacture*, 35(6), 1995.

[Yoda *et al.* 88] H. Yoda, Y. Ohuchi, Y. Taniguchi, and M. Ejiri. An automatic wafer inspection system using pipelined image processing techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(1), January 1988.

[Zeid 91] I. Zeid. *CAD/CAM theory and practice.* McGraw-Hill, Inc., 1991.

[Zhang 93] Z. Zhang. Iterative point matching for registration of free-form curves. Research report number 1658, INRIA, February 1993.

[Zuk & Dell'eva 83] D. M. Zuk and M. L. Dell'eva. Three-dimensional vision system for the adaptive suspension vehicle. Final report no. 170400-3-f, ERIM, DARPA 4468, Defence Supply Service, Whashington, D.C., January 1983.

Appendix A

Glossary of mathematical symbols and abbreviations

This appendix describes all the symbols and abbreviations used in the thesis. The abbreviations and symbols are listed in alphabetic order in Sections A.1 and A.2.

A.1 Abbreviations

1. AIS: application interface.
2. AP: approximated polynomial connection matrix.
3. B-rep: boundary representations.
4. CAD: computer aided design.
5. CAM: computer aided manufacture.
6. CMM: coordinate measuring machine.
7. CNC: coordinate numerical controlled machine.
8. C.Pos.: canonical position.
9. C.R.F.: canonical reference frame.
10. CSG: constructive solid geometry.
11. DFT: 2D Discrete Fourier Transform.

12. DRF: datum reference frame.

13. DSP: Digital Signal Processing.

14. DT: dimension and tolerancing.

15. EDT: Evaluated Dimension and Tolerancing.

16. EL: entity linking node.

17. FFT: Fast Fourier Transform.

18. GDP: a CSG modeller from IBM.

19. GEOTOL: system for tolerance analysis.

20. ICP: iterative closest point algorithm for registragion.

21. ISO and ANSI: tolerancing standards.

22. ORF: object reference frame.

23. PDES: Product Data Exchange Specification.

24. SRF: sensor reference frame.

25. STEP: Standard for the Exchange of Product model data.

A.2 Symbols

1. α: angle between two directions or constant relating the threshold value τ_1 and module of DFT of deterministic deformation.

2. $(\alpha\ \beta\ \gamma)$: define the orientation of the image frame axes.

3. a: one of the three constants that define a line or a plane.

4. a^*: optimal value of a found in least square approximation.

5. a': one of the three constants that define a line or a plane.

6. a_{AB}: one of the three constants defining $Line_{AB}$.

7. a_{AC}: one of the three constants defining $Line_{AC}$.

8. A: 3D edge point corresponding to the intersection of at least 3 planes.

9. Av: percentage of points detected in reconstruction of deformation.

10. AB: segment.

11. AC: segment.

12. $ABCD$: plane in Figure 4.46.

13. $ABEGF$: plane in Figure 4.46.

14. AF: segment.

15. A_i: auxiliary matrix used in the calculation of rotation in Section 2.3.3.

16. A_i^T: transpose of A_i.

17. $Axis_1$, $Axis_2$ and $Axis_3$: main axis of ellipsoid.

18. a_z: main axis of the uncertainty ellipsis associated to feature extraction.

19. β: angle between two directions or security factor used in the determination of size of observation window.

20. b: one of the three constants that define a line or a plane.

21. b^*: optimal value of b found in least square approximation.

22. b': one of the three constants that define a line or a plane.

23. b_{AB}: one of the constants defining $Line_{AB}$.

24. b_{AC}: one of the three constants defining $Line_{AC}$.

25. b_y: main axis of the uncertainty ellipsis associated to feature extraction.

26. B: 3D edge point corresponding to the intersection of at least 3 planes.

27. B_{bin}: defined by Equation (4.4.70).

28. B_c: auxiliary matrix used in the calculation of rotation in Section 2.3.3.

29. B_{lsq}: auxiliary matrix in lest square approximation of plane.

30. c: one of the three constants that define a line or a plane.

31. c^*: optimal value of c found in least square approximation.

32. c': one of the three constants that define a line or a plane.

33. c_{AB}: one of the constants defining $Line_{AB}$.

34. c_{AC}: one of the three constants defining $Line_{AC}$.

35. C: 3D edge point corresponding to the intersection of at least 3 planes

36. C_i: cuboid approximating region data surface region associated with data point $\mathbf{p_i}$.

37. C_{lsq}: auxiliary matrix in the lest square approximation of a plane.

38. ${C_{lsq}}^T$: the transpose of C_{lsq}.

39. cor_{yz}: element in the variance matrix of errors of feature extraction.

40. $\hat{cor}_{yz}$: estimate of element in the variance matrix of errors of feature extraction.

41. **C(P,X)**: operator that finds the set of points **Y** in **X** such that each point in **Y** is the closest point in **X** to a point in **P**.

42. C_0: discontinuity in the range value.

43. C_1: discontinuity on the normal direction.

44. d: is the value of the least square error associated with the registration between two sets of corresponding points.

45. δ_i: standard deviation associated with the η_i - see Equation (5.3.3).

46. δ_{LIM}: limit value of error in feature extraction.

47. $\delta Pose$: random variable corresponding to the error in estimation of pose.

48. $\delta_r(x, y)$: measured difference between the nominal and the manufactured surface (residual image).

49. $\hat{\delta_r}(x, y)$: reconstructed residual image - see Equation (6.5.32).

50. δ_r^{new}: new residual image obtained after subtraction of estimate of deterministic deformation.

51. δy: a value of increment

52. Δ: is the size of the pixel in the y or x direction.

53. Δ_{angle}: increment value.

54. ΔA: area of pixel.

55. Δ_i^z: size of the cuboid C_i on the z direction.

56. Δ_{pose}: increment value.

57. $\Delta Pose^i$: size of subset i of R_P

58. $\Delta Prob^i$: probability of right value of pose being inside the subset i of R_P.

59. Δ_x and Δ_y: sizes of data sampling grid on the x and y directions respectively.

60. d: value of the least square error associated with the registration.

61. d_{AV}: average least square distance.

62. d_f^{max}: maximum distance of the uncertainty region for a probability P_f^{feat} of the actual value being outside the uncertainty region.

63. d_i: distance between face i of polyhedral and origin.

64. d_{ij}: distance between the location i and the location j

65. d_i^{max}: maximum distance between a point inside an uncertainty region and the center of the uncertainty region.

66. d_k: indicates the value of d at iteration **k**.

67. $\mathbf{dir}_{\mathbf{line}}$: vector parallel to the line corresponding to the intersection of two planes.

68. d_n^{max}: is the maximum distance of the uncertainty regions for a probability P_n^{feat} of the actual value being outside the uncertainty region.

69. D_M: average distance between model points and data points - see Equation (5.3.4).

70. d_p and d_n: offset distances.

71. D_s: significance of statistical test.

72. D_1 and D_2: period of deformation in model of waviness error - see Figure (6.5).

73. $D(\mathbf{J}, \mathbf{V})$: distance measure minimised during registration.

74. $\mathbf{D}(x, y)$: z coordinate of $\mathbf{I}(x, y)$.

75. ϵ_a, ϵ_b, $\epsilon_{a'}$ and $\epsilon_{b'}$: errors in the estimates of a, b, a' and b' respectively.

76. ϵ_{conv}: is a small real number used to define the convergence in an iterative algorithm.

77. e_k: cost function to be minimised in Section 2.3.5 to determine registration.

78. $\mathbf{e}(x, y)$: 3D random vector modelling the sensor errors.

79. **ED-form**: tolerance value.

80. EDGE-VERTICE: type of datum that indicates the use of a manufactured line and a manufactured vertex in the determination of datum.

81. **ED-position**: tolerance value.

82. **ED-size**: tolerance value.

83. E_M: average square distance between model points and data points - see Equation (5.3.4).

84. $\mathbf{E}[\cdot]$: the expectation operator.

85. $E(u, v)$: DFT of deterministic deformation.

86. ϕ: angle defining the angle between two directions.

87. ϕ^m: angle used to define the direction of a vector in $Pose_m$.

88. ϕ_p^i: angle used to determine a direction (defined in Equations (4.6.80) to (4.6.85)).

89. ϕ_1 and ϕ_2: angles between the normals of imaged planes and the viewing direction.

90. f_a: actual feature on manufactured object.

91. $f_{\delta Pose}(.)$: probability density function of the random variable $\delta Pose$.

92. $f_{distance}(.)$: cost function minimised to determine edge point.

93. f_g: global feature.

94. $\mathbf{f_i}$: face $\mathbf{i}$ of a polyhedral.

95. f_i^a: feature belonging to the set of manufactured (actual) features F_a.

96. f_i^m: feature belonging to the set of measured features F_m corresponding to the actual feature f_i^a.

97. f_i^m: measured feature i.

98. f_N: nominal feature corresponding to f_a.

99. $f(\mathbf{j})$: correspondence mapping function.

100. f_x and f_y: first derivatives of the object surface on the directions of x and y respectively.

101. F_a: set of manufactured (actual) features.

102. FACE: type of datum that indicates the use of a manufactured planar surface in the determination of datum.

103. F_m: set of measured features.

104. F_{out}: maximum acceptable number of measured features outside the test tolerance zone.

105. $\mathbf{F(O, Par)}$: equation relating vector of observables $\mathbf{O}$ and vector of parameters $\mathbf{Par}$.

106. γ: angle between two directions.

107. $g(m)$: cost function.

108. $g(\ )$: function relating the vector of model parameters to the vector of tolerated dimensions.

109. $g(\mathbf{Par})$: cost function minimised to determine vector of parameters **Par**.

110. $\mathbf{G}(x, y)$: 3D model point corresponding to the 3D image point $\mathbf{I}(x, y)$.

111. $\hat{\eta}$: reconstructed random process η - see Equation (6.5.32).

112. η_i: isotropic random vector process normally distributed that models the sensor and modelling errors.

113. ${\eta_i}^T$: transpose of η_i.

114. $\eta_{sensor}(x, y)$: error created by sensor errors in the residual image.

115. η_i': anisotropic random vector process normally distributed that models the sensor and modelling errors.

116. ${\eta_i'}^T$: transpose of η_i'.

117. $\eta(.,.)$: normally distributed random process.

118. $h(\ )$: function relating the vector of tolerated dimensions to the vector of model parameters.

119. H: amplitude of deformation in model of waviness error - see Figure (6.5).

120. H_{min}: minimum amplitude of deformation that must be detected.

121. H_0 and H_1: statistical hypotheses.

122. i: integer number.

123. $Im(u, v)$: imaginary part of DFT of residual.

124. I_p^{global}: diagnosis of the inspection of a global feature.

125. I_p^{pose}: diagnosis in pose inspection.

126. I_p^{shape}: diagnosis of shape inspection.

127. I_p^{size}: diagnosis of size inspection.

128. $I_p(F_m, T_{test}, Pose, Scale, F_{out})$: result of inspection of set of features F_m using the test tolerance zone $T_z^{test}(T_{test}, Pose, Scale)$ when the maximum acceptable number of measured features outside the test tolerance zone is equal F_{out}.

129. $\mathbf{I}_r(x, y)$: 3D point of actual range image (with sensor errors).

130. $\mathbf{I}(x, y)$: 3D point of ideal range image (without sensor errors).

131. j: integer number.

132. $\mathbf{j}$: 3D point on a surface.

133. $\mathbf{j_i}$: 3D point on view $\mathbf{J}$ of surface.

134. $\mathbf{j_i'}$: 3D point on view $\mathbf{J}$ of surface.

135. $\mathbf{J}$: view of a surface.

136. k: integer number.

137. k_1 and k_2: constants defining parametric model of edge.

138. k_1^* and k_2^*: values of k_1 and k_2 at minimum of $f_{distance}$.

139. K_{angle}: auxiliary variable in the evaluation of error of vertex extraction.

140. K_c: process capability index constant.

141. K_{ON}: number of peaks of the DFT of the residual that are bigger than τ_1.

142. K_{scale}: scale factor.

143. K_0: scale factor.

144. $K(u, v)$: function defining the peaks of the DFT of residual image that are bigger than τ_1 - see Equation (6.5.32).

145. λ_1, λ_2 and λ_3: eigenvalues of *3* X *3* matrix.

146. l: integer number.

147. l^2: define the area of deformation in model of waviness error- see Figure (6.5).

148. L: least material condition.

149. L_d: minimum size of window the deformation must cover.

150. $Line_{AB}$: least square approximations of segment AB.

151. $Line_{AC}$: least square approximations of segment AC.

152. L_{lim}: limit value of a variable.

153. $\mathbf{m}$: vector of model parameters.

154. $\mathbf{m_i}$: unit vector parallel to normal direction at point $\mathbf{p_i}$ according to model. be detected.

155. m_R and m_I: see Equation (6.3.23).

156. $\mathbf{M}$: rigid transformation expressed as a 4 X 4 matrix.

157. M: module of the DFT of the deterministic deformation ($E(u, v)$).

158. M: maximum material condition.

159. $\mathbf{M}(u, v)$: 3D point on the model of a manufactured sculptured surface.

160. N: integer number used to count something.

161. N_b: number of points in P_{big}.

162. $\mathbf{n_i}$: measured unit vector parallel to the surface normal at the point $\mathbf{p_i}$.

163. $\mathbf{n}(u, v)$: normal vector at the point $\mathbf{M}(u, v)$.

164. N_{feat}: number of features.

165. N_p: number of 3D points in P_{plane}.

166. N_{pixels}: number of points in P_{small} and P_{edge} when these numbers are equal.

167. N_{pose}: define the number of elements in search grid.

168. $N_\rho d\rho$: number of points in the infinitesimal deformed region for which $\hat{\epsilon_d}(x, y) = \rho$.

169. N_s: number of points in P_{small}.

170. N_T: total number of realisations in Monte Carlo simulation.

171. N_{sq}: square root of N_p.

172. NULL: indicates a null kind or type of datum.

173. $\mathbf{O}$ is a vector of observables extracted from the object image.

174. ONE-FACE: kind of datum define by one face.

175. ONE-LINE: kind of datum defined by one line.

176. $\mathbf{O}(u, v)$: 3D point on manufactured sculptured surface.

177. p_i: 3D point in range image.

178. $\mathbf{p_i}$: measured point on surface of object.

179. P: value of probability.

180. $\mathbf{P}$: set of n data points $\{\mathbf{p_i}\}_{\mathbf{i=1,n}}$.

181. P_A: probability associated to uncertainty region in the extraction of vertex A.

182. $\hat{P}_A$: normal estimate of P_A.

183. **Par**: vector of parameters corresponding to the pose of an object.

184. P_{big}: set of 3D points (subset of P_{edge}).

185. P_{bin}: defined by Equation (4.4.71).

186. $P_{edge} = \{p_1, p_2, ..., p_N\}$: set with N 3D points in the same column (or row) of range image.

187. P_f: probability of rejecting a manufactured element that was manufactured within the designed tolerance

188. P_f^{dft}: maximum acceptable probability of false alarm during the detection of periodic deformations.

189. P_f^{feat}: probability of a measured feature f_i^m to be outside the test tolerance zone when the actual feature f_i^a is inside of the designed tolerance zone.

190. P_f^H: maximum acceptable value of the probability of rejecting the hypothesis H_0 when H_0 is true.

191. $P_f{}^{min}$: minimum value of probability of false alarm in a very sensitive test for a given ratio $\frac{S}{N}$.

192. P_f^{test}: probability of false alarm in statistical test for detecting deterministic deformation.

193. $P_f^{test}(T_H)$: probability of false alarm in Moran's statistical test with test value equal to T_H.

194. P_i: probability associated to uncertainty region R_i.

195. $\hat{P}_i$: normal estimate of the probability P_i.

196. $\mathbf{P}_{ij}$: control point in B-spline.

197. $\mathbf{P_k}$: indicates $\mathbf{P}$ at iteration $\mathbf{k}$.

198. P_n: probability of not detecting that the element being inspected was manufactured out of the conventional tolerance.

199. P_n^{dft}: maximum acceptable probability of not detecting a periodic deformation.

200. P_n^{feat}: probability of a measured feature f_i^m to be inside the test tolerance zone when the actual feature f_i^a is outside of the conventional tolerance zone.

201. $Pose_0$ and scale $Scale_0$: pose and scale associated to the least square approximation of the set of measured features.

202. $Point_{AB}$: set of 3D points belonging to segments AB.

203. $Point_{AC}$: sets of 3D points belonging to segment AC.

204. P_{OK}: value of probability.

205. P_{out}: probability of the right value of pose being outside the uncertainty region R_P or maximum acceptable probability of point on the surface being manufactured out of tolerance.

206. $Pose$: pose of element offset to create test tolerance zone.

207. $Pose_{(i,j,k,l,w)}$: pose in search grid - see Equation (4.6.80).

208. $Pose_m = [x_p^m, y_p^m, z_p^m, \theta^m, \phi^m]$: pose of least square approximation of the set of measured features.

209. $P_{plane} = \{p_1, p_2, ..., p_{N_p}\}$: set of N_p 3D points belonging to a plane.

210. P_{small}: set of 3D points (subset of P_{edge}).

211. P_1: maximum acceptable probability of a point not belonging to the deformation being classified as in the deformation.

212. P_2: maximum acceptable probability of a point belonging to the deformation not being detected.

213. $\mathbf{P(U_i)}$: object surface (plane or quadric) **i**.

214. $\{\mathbf{P(U_i)}\}_{\mathbf{i=1,n}}$: set of n surfaces.

215. θ: angle defining the angle between a line and a reference frame axis.

216. θ_p^i: angle used to determine a direction (defined in Equations (4.6.80) to (4.6.85)).

217. θ^m: angle used to define the direction of a vector in $Pose_m$.

218. $\mathbf{\Theta}$: angle used to define a rotation **R**.

219. **q**: vector containing the rotation and translation of rigid transformation.

220. $\mathbf{q_k}$: indicates the value of **q** at iteration **k**.

221. Q_v: auxiliary matrix used in the calculation of rotation in Section 2.3.3.

222. **Q(P,Y)**: operator that returns the rigid transformation aligning the sets of point **P** and **Y** and the least square distance between the corresponding points of the two sets after alignment.

223. ρ: value of the reconstructed deformation.

224. ρ_{max}: maximum value of ρ.

225. ρ_{min}: minimum value of ρ.

226. r: distance of the pixel (x, y) in the deformed window to the center of the deformed window.

227. $\mathbf{r}$: vector used to define a rotation $\mathbf{R}$.

228. $radius$: radius of circle.

229. $\hat{radius}$: estimate of $radius$.

230. r_{dist}: range measurement in radar.

231. $\mathbf{R}$: indicates a rotation.

232. $Re(u, v)$: real part of DFT of residual.

233. R_g: uncertainty region of global measured feature.

234. R_i: uncertainty region associated to measured feature i.

235. R^k: defined in Equations (4.6.87) to (4.6.89).

236. R_{lim}: size of uncertainty region in estimate of hole's center.

237. R^M: radius of circle obtained during the extraction of the edge points of a hole.

238. R_P: uncertainty region associated to pose determination.

239. $R(u, v)$: DFT of residual image -see Equation (6.3.7).

240. σ_η: standard deviation of $\eta(.,.)$.

241. $\hat{\sigma_\eta}$: estimate of σ_η.

242. $\sigma_{\hat{\eta}}$: standard deviation of $\hat{\eta}$.

243. σ_m: standard deviation of the manufacturing process.

244. σ_{max}: maximum acceptable value of standard deviation of manufacturing process.

245. σ_S: standard deviation of error in range measurement.

246. s: variable used to parameterise a surface.

247. $\mathbf{s_i}$: indicates a duple composed by a measured point on the surface of an object and a measured unit vector parallel to the surface normal at this point.

248. *step*: size of C_0 discontinuity in step edge.

249. s_x: size of pixel in x direction.

250. s_y: size of pixel in y direction.

251. S: regardless of material condition.

252. *Scale*: scale of element offset to create test tolerance zone.

253. S_f: subset of the uncertainty region R_P.

254. $\mathbf{S_i^k}$: tangent plane $\mathbf{i}$ of surface $\mathbf{V}$ found during the iteration $\mathbf{k}$.

255. $\{\mathbf{s_i} = (\mathbf{p_i}, \mathbf{n_i})\}_{i=1,m}$: set with m elements $\mathbf{s_i}$ used by [Grimson & Lozano-Perez 84] for matching.

256. *Size*: size of element offset to create test tolerance zone.

257. S_N: nominal scale of the geometrical element being inspected.

258. τ: time interval between emission and reception of signal by radar.

259. τ_1: threshold used during the detection of periodic deformations.

260. τ_2: threshold used during the reconstruction of periodic deformations.

261. t: variable used to parameterise a surface.

262. $\mathbf{t}$: vector of tolerated dimensions.

263. T: shape parameter that allows to change the shape of the deformation.

264. $\mathbf{T}$: translation in rigid transformation.

265. $\hat{\mathbf{T}}$: estimate of translation $\mathbf{T}$.

266. THREE-FACES: kind of datum defined by three planes.

267. T_{out}: minimum number of features outside the conventional tolerance zone detectable with probability $1 - P_n$.

268. T_{test}: size of test tolerance zone used during inspection.

269. TWO-EDGES: type of datum that indicates the use of two manufactured lines in the determination of datum.

270. THREE-VERTICES: type of datum that indicates the use of three manufactured vertices in the determination of datum.

271. T_S: uniform knot vector in B-spline.

272. TWO-VERTICES: type of datum that indicates the use of two manufactured vertices in the determination of datum.

273. TY1, TY2, TY3: indicate the type of the datums T1, T2 and T3 respectively.

274. T_z: size of conventional tolerance zone.

275. T_z^m: size of designed tolerance zone.

276. T_z^{max}: maximum possible value of the size of tolerance zone.

277. T_z^{min}: minimum possible value of the size of tolerance zone.

278. $T_z^{test}(T_{test}, Pose, Scale)$: test tolerance zone of size T_{test} such that the geometrical element offset to create the tolerance zone has pose $Pose$ and scale $Scale$.

279. T1, T2, and T3: indicate the datums of a DRF node.

280. $\mathbf{T}(.)$: operator indicating a rigid transformation (translation and rotation).

281. u: variable parameterising sculptured surface or DFT of residual image.

282. u_x^i, u_y^i, u_z^i: elements of the vector equal to the sum between two surface normals.

283. $\mathbf{U_i}$: vector of parameters that defines the surface $\mathbf{i}$.

284. v: variable parameterising sculptured surface or DFT of residual image.

285. $\mathbf{v}$: 3D point on a surface.

286. var_{xx} and var_{yy}: elements in the variance matrix of errors of feature extraction.

287. $\hat{var}_{xx}$, $\hat{var}_{yy}$: estimates of elements in the variance matrix of errors of feature extraction.

288. $\mathbf{v_i^k}$: point $\mathbf{i}$ of surface $\mathbf{V}$ found during the iteration $\mathbf{k}$.

289. V: speed of signal propagation in radar.

290. $\mathbf{V}$: view of a surface.

291. Var: variance matrix of error in the determination of pose.

292. Var_A: variance matrix of errors in the extraction of vertex A.

293. v_x^i, v_y^i, v_z^i: elements of the vector equal to the difference between two surface normals.

294. $V(.)$: see Equation (6.3.20).

295. x: coordinate in the x direction.

296. x_{min}: coordinate in the x direction.

297. x_{max}: coordinate in the x direction.

298. x_A: coordinate of the vertex A in the x direction.

299. $\hat{x_A}$: estimate of x_A.

300. x_{center}: x coordinate of hole's center.

301. $\hat{x_{center}}$: estimate of x coordinate of hole's center.

302. x_{center}^i: defined in Equations (4.6.87) to (4.6.89).

303. x_{center}^M: x coordinate of the center of circle obtained during the extraction of the edge points of a hole.

304. $\mathbf{x_i}$: model point.

305. $\mathbf{x}_{i,j}$: 3-D vector.

306. $\mathbf{x_i}^k$: closest model point to data point $\mathbf{p_i}$ at iteration k.

307. x_{left}: x coordinate of pixel in range image.

308. $x_m(u,v)$: x coordinate of $\mathbf{M}(u,v)$ in the ORF.

309. x_o: coordinate of 3D point on the x direction.

310. $\hat{x_o}$: estimate of x_o.

311. $x_o(u,v)$: x coordinate of $\mathbf{O}(u,v)$ in the ORF.

312. x_p: x coordinate of a 3D point.

313. x_p^i: 3D point defined in Equations (4.6.80) to (4.6.85).

314. x_p^m: define the x coordinate of a 3D point in $Pose_m$.

315. x_{right}: x coordinate of pixel in range image.

316. x_0: x coordinate of a 3D or 2D point.

317. **x**(s, t) : parameterised surface.

318. **X** is a model shape.

319. X_{lsq}: solution of least square approximation of plane.

320. w: integer number.

321. $w_{i,j}$: real number.

322. W_d: minimum possible size of observation window.

323. W_N: see Equation (6.3.7).

324. W_o: size in pixels of side of observation window.

325. W_s: size in pixels of side of residual image.

326. y: coordinate in the y direction.

327. y_A: coordinate of the vertex A in the y direction.

328. $\hat{y_A}$: estimate of y_A.

329. y_{center}: y coordinate of hole's center.

330. y_{center}^j: defined in Equations (4.6.87) to (4.6.89).

331. $y_{\hat{center}}$: estimate of y coordinate of hole's center.

332. y^{M}_{center}: y coordinate of the center of circle obtained during the extraction of the edge points of a hole.

333. y_i: y coordinate of the pixel p_i in P_{edge}.

334. $y_m(u,v)$: y coordinate of $\mathbf{M}(u,v)$ in the ORF.

335. y_o: coordinate of 3D point on the y direction.

336. $\hat{y_o}$: estimate of y_o.

337. $y^{\star}_o$: value of y coordinate that minimise $f_{distance}$.

338. $y_o(u,v)$: y coordinate of $\mathbf{O}(u,v)$ in the ORF.

339. y_p: y coordinate of a 3D point.

340. y^i_p: 3D point defined in Equations (4.6.80) to (4.6.85).

341. y^m_p: define the y coordinate of a 3D point in $Pose_m$.

342. y_0: y coordinate of a 3D or 2D point.

343. $\mathbf{Y}$ is a set of n model points $\{\mathbf{x_i}\}_{\mathbf{i=1,n}}$.

344. $Y = \{y^{-M}, ..., y^0, ..., y^M\}$: set of y coordinates used in the minimisation of $f_{distance}$.

345. $\mathbf{Y_k}$: indicates the value of $\mathbf{Y}$ at iteration $\mathbf{k}$.

346. z_A: coordinate of the vertex A in the z direction.

347. $\hat{z_A}$: estimate of z_A.

348. z_{AV}: average value of z coordinate.

349. z_{cte}: absolute value of the constant component of the error of the estimate of z_o.

350. z_i: z coordinate of the point p_i in P_{edge}.

351. $z_m(u,v)$: z coordinate of $\mathbf{M}(u,v)$ in the ORF.

352. z_o: coordinate of 3D point on the z direction.

353. z_o^*: value of z_o at minimum of $f_{distance}$.

354. $\hat{z_o}$: estimate of z_o.

355. $z_o(u,v)$: z coordinate of $\mathbf{O}(u,v)$ in the ORF.

356. z_p: z coordinate of a 3D point.

357. z_p^i: 3D point defined in Equations (4.6.80) to (4.6.85).

358. z_p^m: define the z coordinate of a 3D point in $Pose_m$.

359. z_{random}: absolute value of the random component of the error in the estimate of z_o.

360. z_0: z coordinate of a 3D or 2D point.

Appendix B

Model Validation

This appendix discusses the accuracy of the statistical models built in Chapter 4.

Section B.1 introduces the appendix by describing our strategy to create the statistical models. Section B.2 analyses the accuracy of the model we used to describe the errors of the range measurements in laser stripers. Section B.3 verifies the differences in performance caused by the difference between the actual distribution of the errors in the range measurement and the normal distribution. Section B.4 investigates the variation of the errors of the range measurement with the viewpoint. Section B.5 illustrates the accuracy of our statistical models in some particular situations. Section B.6 summarises the appendix.

B.1 Introduction

In Chapter 4 we used Monte Carlo simulations to create statistical models describing the errors of some feature extraction algorithms used in inspection procedures. Our strategy for the creation of these models was based on the application of the feature extraction operators to synthetic range images.

Therefore, the validity of our statistical models is related to the accuracy in the generation of the synthetic images, *i.e.* the accuracy with which our models describe the errors in the feature extraction processes depends basically on how close to the real range images our synthetically generated images were.

The generation of the synthetic images used in Chapter 4 was done by:

1. **Generation of ideal range measurement**: each pixel in the range image is associated with an ideal value of range measurement that is equal to the average height of the surface over the pixel area.

2. **Corruption of ideal measurement with noise**: a normally distributed random process with mean equal to zero and standard deviation equal to σ_S is added to the ideal range measurement.

The difference between the actual range images and the synthetic range images depends upon a series of factors that we did not consider in the simulation such as: the calibration of the range sensor, the errors caused by non-uniform sampling, the measurement errors near C_0 and C_1 discontinuities on the surface being imaged, interpolation errors, specularities and the errors caused by using a normally distributed random variable in stage 2 of the generation of the synthetic images.

B.2 Model of errors in the range measurement

The description of the errors in the range measurement as a normally distributed random process with mean zero and standard deviation σ_S (used in stage 2 of the generation of the synthetic images) is not exactly correct.

However, this description is not far from reality as illustrated by Figure B.1. The figure shows the probability distribution function of the errors in the range measurement and the approximation of this function using the probability distribution of a normally distributed random process.

Figure B.1 was obtained using Edinburgh's Laser Striper to take an image of a planar surface on a machined part normally used to calibrate the laser striper. The image was taken in such a way that the planar surface was perpendicular to the viewing direction. Furthermore, the accuracy in the manufacturing of the object imaged (that we will call calibration block) was such that all the variations in the range value were essentially due to measurement errors.

Figure B.2 shows the histogram of the residuals in the range image of the planar surface of the calibration block relative to a planar least square fit and its approximation by a

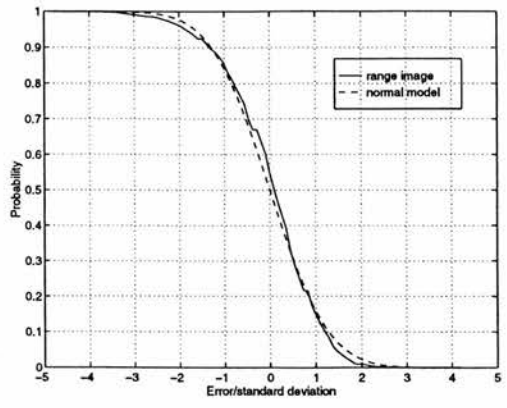


Figure B.1: Distribution of residual from a plane fit in actual range image and its approximation by a normally distributed variable. Result obtained using 8100 range measurements. Estimated value of σ_S equal to *0.1 mm*. Angle between plane's normal and viewing direction equal to *0.0 rad*.

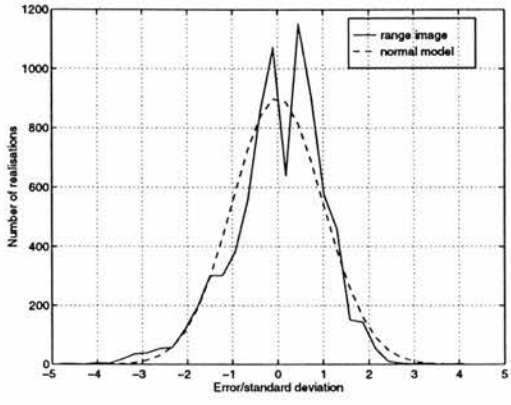


Figure B.2: Histogram of residuals from a plane fit to range image and its approximation using a normally distributed random variable. Result obtained using 8100 range measurements. Angle between plane's normal and viewing direction equal to *0.0 rad*.

normally distributed random variable.

More accurate simulations can be obtained by a careful statistical modelling of the probability distribution function of the errors in the range measurements. In the next section we illustrate this procedure in order to evaluate the differences in performance caused by the use of the actual distribution of the errors in the range measurements instead of a normal approximation of it.

	$\phi_2 = 0.16$ rad	$\phi_2 = 0.47$ rad	$\phi_2 = 0.79$ rad	$\phi_2 = 1.1$ rad
$\phi_1 = 0.0$ rad	2.390	0.670	0.256	0.182
$\phi_1 = 0.31$ rad	1.660	1.260	0.399	0.191
$\phi_1 = 0.63$ rad	0.397	1.219	1.062	0.247
$\phi_1 = 0.94$ rad	0.245	0.280	0.619	0.415

Table B.1: Maximum absolute error in the estimate of y_0 (in mm) for different angles ϕ_1 and ϕ_2 between the intersecting planes and the viewing direction. Result of 1000 realisations for each pair of values of ϕ_1 and ϕ_2. Relative pose of 3D edge lines with relation to sensor created randomly. Synthetic images created considering pixels of $0.5\ mm \times 0.5\ mm$ and actual distribution of noise in error of range measurement. Edge extraction with $N_s = N_b = 18$.

	$\phi_2 = 0.16$ rad	$\phi_2 = 0.47$ rad	$\phi_2 = 0.79$ rad	$\phi_2 = 1.1$ rad
$\phi_1 = 0.0$ rad	0.175	0.145	0.115	0.190
$\phi_1 = 0.31$ rad	0.380	0.420	0.267	0.298
$\phi_1 = 0.63$ rad	0.276	0.430	0.433	0.267
$\phi_1 = 0.94$ rad	0.208	0.445	0.576	0.353

Table B.2: Maximum absolute error in the estimate of z_0 (in mm) for different angles ϕ_1 and ϕ_2 between the intersecting planes and the viewing direction. Result of 1000 realisations for each pair of values of ϕ_1 and ϕ_2 under the same conditions of Table B.1.

B.3 Simulations with actual distribution of noise in error of range measurement

In this section we repeat the simulations done in Section 4.3.1.3 using the actual distribution of the error in the range measurement (shown in Figure B.1) instead of assuming the error normally distributed as we did in Section 4.3.1.3. Our objective is to observe the differences in performance caused by the difference between the actual distribution of the error in the range measurement and the normal distribution.

We start our experiments by repeating the experiments that produced Tables 4.1 to 4.4 using the actual distribution of the error in the range measurement shown in Figure B.1. The results are shown in Tables B.1 to B.4.

The comparison between the two sets of table reveals that:

- there were a considerable variation in performance between Tables 4.1 and Table B.1 for many entries in these tables,
- the differences between Table 4.2 and Table B.2 were smaller than the differences

	$\phi_2 = 0.16$ rad	$\phi_2 = 0.47$ rad	$\phi_2 = 0.79$ rad	$\phi_2 = 1.1$ rad
$\phi_1 = 0.0$ rad	0.532	0.152	0.080	0.044
$\phi_1 = 0.31$ rad	0.341	0.278	0.088	0.046
$\phi_1 = 0.63$ rad	0.114	0.238	0.198	0.053
$\phi_1 = 0.94$ rad	0.060	0.072	0.132	0.091

Table B.3: Standard deviation of error in the estimate of y_0 (in mm) for different angles ϕ_1 and ϕ_2 between the intersecting planes and the viewing direction. Result of 1000 realisations for each pair of values of ϕ_1 and ϕ_2 under the same conditions of Table B.1.

	$\phi_2 = 0.16$ rad	$\phi_2 = 0.47$ rad	$\phi_2 = 0.79$ rad	$\phi_2 = 1.1$ rad
$\phi_1 = 0.0$ rad	0.052	0.079	0.060	0.088
$\phi_1 = 0.31$ rad	0.134	0.099	0.079	0.060
$\phi_1 = 0.63$ rad	0.063	0.143	0.150	0.180
$\phi_1 = 0.94$ rad	0.077	0.128	0.208	0.140

Table B.4: Standard deviation of error in the estimate of z_0 (in mm) for different angles ϕ_1 and ϕ_2 between the intersecting planes and the viewing direction. Result of 1000 realisations for each pair of values ϕ_1 and ϕ_2 under the same conditions of Table B.1.

between Table 4.1 and Table B.1 and

- the results in Tables 4.3 and 4.4 are very close to the results in Tables B.3 and B.4.

By comparing Tables 4.1 and and Table B.1 we can observe a considerable variation in performance for many entries in the tables. However, we also observed that the biggest errors in both sets of tables happened to the same values of angles ϕ_1 and ϕ_2. This fact indicates that the algorithm had the same qualitative behaviour in the two sets of tables.

It is important to observe that the differences between Tables 4.1 and 4.2 and Tables B.1 and B.2 are not as important as the differences between the Tables 4.3 and 4.4 and Tables B.3 and B.4. This happens because in the first case the differences correspond only to the difference in one particular run of the simulation (the run with biggest error) while the differences in the second case take into account all the runs of the simulation.

Figures B.3 to B.6 show the variation in the performance of the algorithm with the value of N_{pixel} for a given viewpoint condition. These figures correspond to the Figures

4.5 to 4.8 respectively and the only difference in the simulation conditions used to obtain the two sets of figures is the kind of distribution used for the errors in the range measurement. In the case of Figures B.3 to B.6 the actual distribution of errors was used and in the case of Figures 4.5 to 4.8 a normal distribution was used.

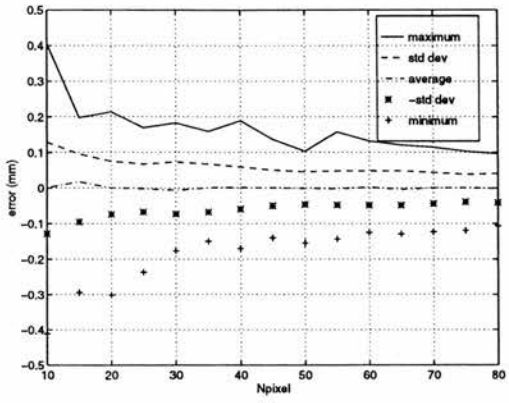


Figure B.3: Maximum error, standard deviation, average error, minus standard deviation and minimum error of estimate of y_o. Result of 300 simulations to each value of N_{pixel} for $\phi_1 = 0.8$ *rad*, $\phi_2 = 1.0$ *rad*, 2D edge line making angle of 45° with x direction, 3D edge line making angle of *0.8 rad* with the image plane and N_{pixel} varying from 20 to 80 pixels. Simulation using actual distribution of error in range measurement.

The comparison between the Figures B.3 to B.6 and the Figures 4.5 to 4.8 reveal a few observations:

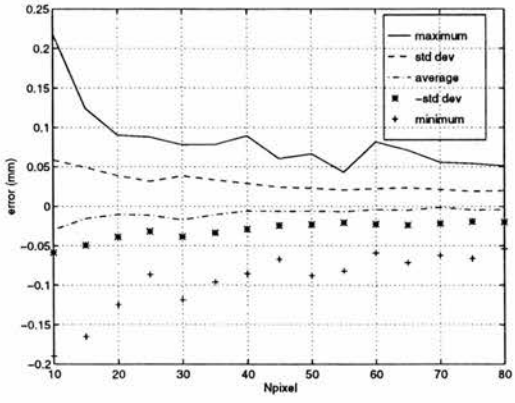


Figure B.4: Maximum error, standard deviation, average error, minus standard deviation and minimum error of estimate of z_o. Result of 300 simulations under the same conditions of Figure B.3.

- the average error of the algorithm in Figures B.3 and B.4 is close to zero in the same way as in Section 4.3.1.3,

- the maximum error and the standard deviation of the errors in Figures B.3 and B.4 decrease as the value of N_{pixel} increases in the same way as in Section 4.3.1.3 and

- the variance and the covariance of the errors decrease in Figure B.5 as the value of N_{pixel} increases in the same way as in Section 4.3.1.3.

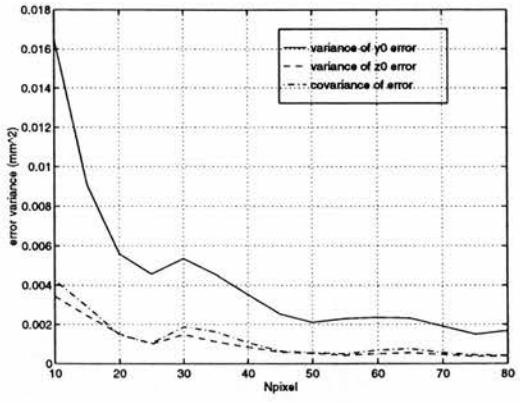


Figure B.5: Variance of the error in the estimate of y_o, variance of the error in the estimate of z_o and covariance between the errors in the estimates of y_o and z_o. Result of 300 simulations under the same conditions of Figure B.3.

From all these facts we can conclude that the algorithm presented the same qualitative behaviour with both kinds of distributions of the errors in the range measurement. Despite this fact, however, there was a variation in the numerical results obtained in both simulations which indicates a considerable variation in performance as we can see by comparing Figure 4.7 to Figure B.5 and Figure 4.8 to Figure B.6. The comparison between these two sets of figures shows that the standard deviation of the errors in the case of the actual distribution of errors in the range measurement is considerably smaller.

Figures B.7 and B.8 show the errors involved in approximating the errors of the roof extraction algorithm by a normally distributed random vector when we have $N_{pixel} = 40$ for the same simulation conditions as in Figure B.3. As in Section 4.3.1.3 (that correspond to the Figures 4.10 and 4.11 respectively) the normal approximation is not precise and underestimates the actual probability associated with the error in the roof edge extraction. This means that Figures B.7 and B.8 illustrate the same qualitative behaviour illustrated by Figures 4.10 and 4.11.

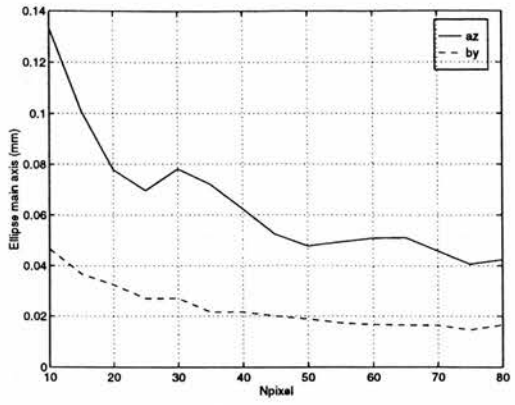


Figure B.6: Main axis of ellipse limiting uncertainty region of extraction algorithm. Result of 300 simulations under the same conditions of Figure B.3.

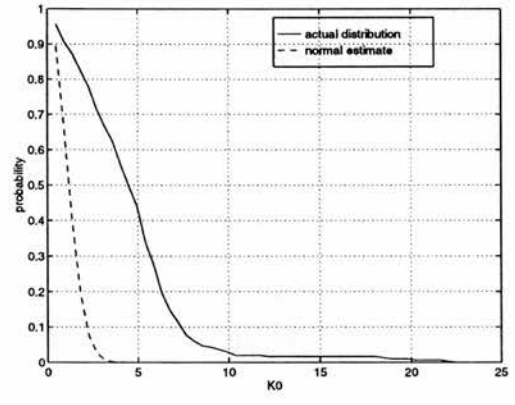


Figure B.7: Probability P_i and normal estimate $\hat{P}_i$ of probability P_i. Result of 300 simulations under the same conditions as Figure B.3 when N_{pixel} is equal to 40 pixels.

The comparison between the results shown in Section 4.3.1.3 and the results found in this section allows us to conclude that the approximation of the error in the range measurement by a normally distributed random variable yields the same qualitative behaviour as the behaviour observed with the actual distribution of error in the error measurement. However, we can also conclude that accurate estimates of performance can only be obtained using the actual distribution of the error in the range measurement.

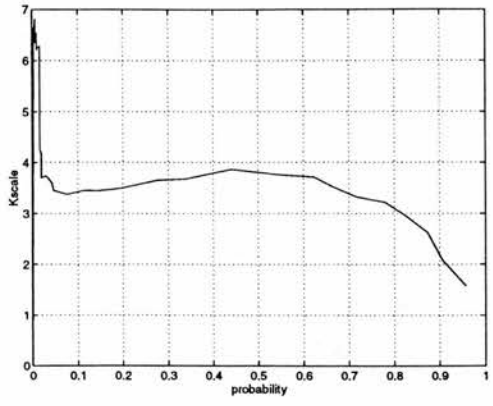


Figure B.8: Variation of K_{scale} with P_i. Result of 300 simulations under the same conditions of Figure B.3 when N_{pixel} is equal to 40 pixels.

B.4 Variation of errors in range measurement with viewpoint

In this section we assess the variation of the errors in the range measurements according to the viewpoint conditions. In order to do so, we repeat the experiments of Section B.2 using planes whose normal makes an angle different from 0° with the viewing direction. Our main objective is to verify if the errors increase when the angle between the plane's normal and the viewing direction increases.

The procedure adopted in our experiments was:

1. Range image acquisition.
2. Least square fitting of plane to range image - see Equations (4.3.47) to (4.3.50).
3. Calculation of the difference between plane estimate and actual range measurement for all image points considered (difference between the z coordinates of the plane estimate and the z coordinates of range image).

Angle (rad)	Estimated σ_S (mm)
0.00	0.10
0.45	0.11
0.96	0.12

Table B.5: Estimated value of standard deviation (σ_S) for different angles between the plane's normal and the viewing direction.

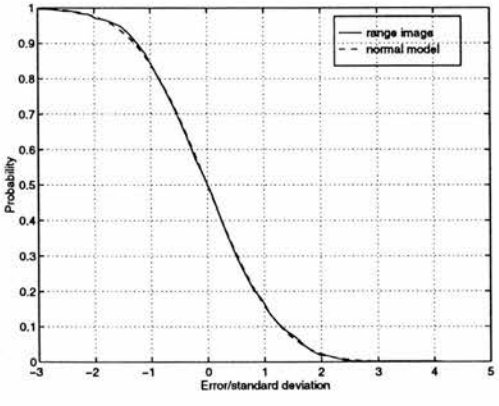


Figure B.9: Distribution of residual from plane fit in actual range image and its approximation by a normally distributed random variable. Result obtained using 592 range measurements. Estimated value of σ_S equal to *0.11 mm*. Angle between plane's normal and viewing direction equal to *0.45 rad*.

The results obtained in the experiment can be summarized by Table B.5 and Figures B.1, B.2, B.9, B.10,B.11 and B.12. Figures B.1, B.9 and B.11 show the distribution of errors in actual range images. Figures B.2, B.10 and B.12 show the histogram of errors in the range measurements. The figures show the results obtained when we vary the angle between the normal of the plane being imaged and the viewing direction.

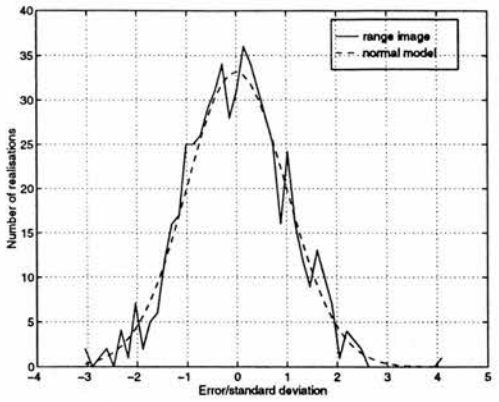


Figure B.10: Histogram of residuals from a plane fit to the range image and its approximation using a normally distributed random variable. Result obtained using 592 range measurements. Angle between plane's normal and viewing direction equal to *0.45 rad.*

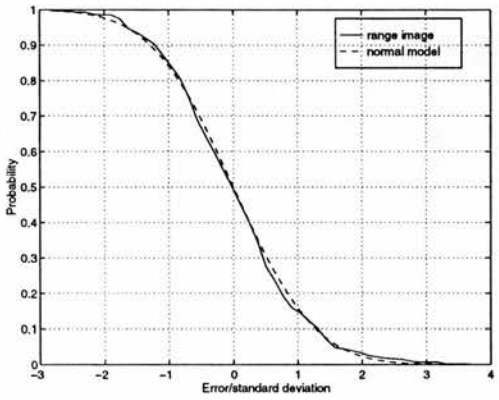


Figure B.11: Distribution of residual from a plane fit in actual range image and its approximation by a normally distributed variable. Result obtained using 560 range measurements. Estimated value of σ_S equal to *0.12 mm.* Angle between plane's normal and viewing direction equal to *0.96 rad.*

From the results observed in Table B.5 we can conclude that the errors in the range measurement did not present a considerable variation for the planes considered (the estimated value of σ_S did not change much from entry to entry in the table). Furthermore, Figures B.1, B.2, B.9,B.10, B.11 and B.12 also did not show any significant variation in performance. Therefore, we can conclude that for the range of angles considered (angles smaller than *0.96 rad*) there is not a great variation in the error of the range measurements with the angle between the viewing direction and the plane's normal. However, it is important to observe that Figure B.12 makes clear that there is a significant systematic component to the real distribution.

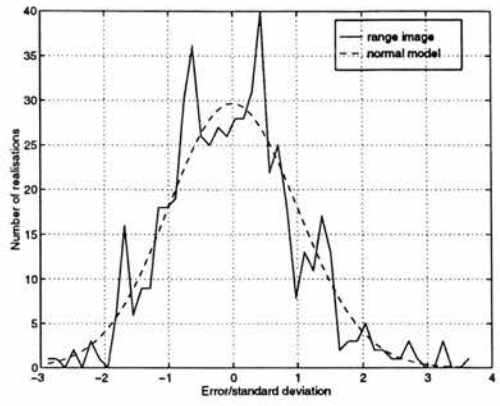


Figure B.12: Histogram of residuals from a plane fit to the range image and its approximation using a normally distributed random variable. Result obtained using 560 range measurements. Angle between plane's normal and viewing direction equal to *0.96 rad.*

B.5 The accuracy of the statistical models

The final effect in the accuracy of the statistical models of errors in the extraction of 3D edge points is illustrated in Figures B.13 and B.16.

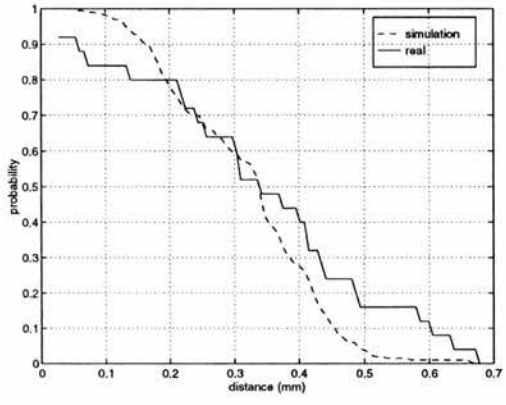


Figure B.13: Density probability function of errors in 3D edge points extracted from range image and its approximation obtained by simulation. Edge points extracted from range image of segment AD (see Figures B.15 and B.14) using $N_{pixel} = 8$.

Figure B.13 shows the probability distribution function of the errors in the extraction of the roof edge points of the range image of segment AD (see Figures B.15 and B.14), as well as the distribution of the same errors according to the result of 300 simulations. The figure was obtained by calculating the distance between the 3D edge points extracted from the range image and an infinite line positioned in such a way that the maximum distance between the edge points and the line was minimised.

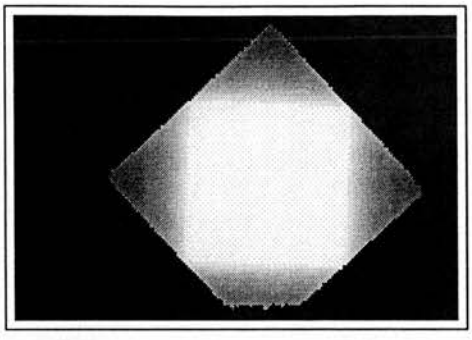


Figure B.14: Range image of the *calibration block*: 200 × 137 pixels of size *1 mm* × *1 mm* obtained using a sensor with $\sigma_S = 0.1$ *mm*.

As we can see in Figure B.13, the result of the simulation is not very close to the reality. Despite this fact the the errors at the tail of the distribution are not extremely big. As a consequence of this fact, the simulations can be used to estimate the maximum error in the roof extraction for low values of probability with an error of approximately *30* %.

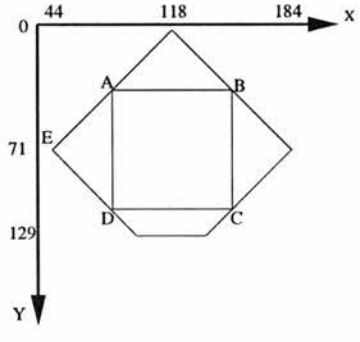


Figure B.15: Projection on image plane of range image of Figure B.15.

Figure B.16 shows the projection of the 3D step edge points extracted from the hole in the range image of Figure 4.45 (after the cleaning operation described in Section 4.6.3) and the corresponding 3D edge points extracted from a synthetic image of the hole. Because the position of the extracted 3D edge points is practically the same for different realisations of the synthetic image, Figure B.16 is a good illustration of the typical accuracy of the simulation.

As we can see from the figure, the results with the synthetic image are very close to the results with the actual image of the hole, despite the fact that the simulation did not take into account the cleaning operation.

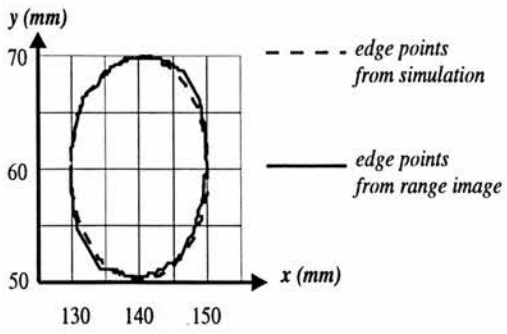


Figure B.16: 3D edge points extracted from range image of hole in Figure 4.45 and corresponding 3D edge points extracted from synthetic range image of hole. Edge points extracted using $N_{pixe} = 10$.

Although Figures B.13 and B.16 show a reasonable agreement between reality and simulation, more accurate models could be obtained by a more accurate modelling of the behaviour of the laser striper and the effects of the algorithms.

B.6 Conclusions

In this appendix we discussed the accuracy of the statistical models that we built in Chapter 4.

The accuracy of the statistical models was related to the errors in the simulation process. The errors in the simulation process, as well as their influence in the final accuracy in the modelling of the errors of the feature extraction were analysed. Results with actual range images were used to show the accuracy of the statistical models of Chapter 4.

From the results of the experiments done in this appendix we can conclude that the use of synthetic images (generated as described in Section B.1) does not produce accurate estimates of the perfomance of the feature extraction algorithms. This means that the synthetic images can only be used to observe the qualitative behaviour of the feature extraction algorithm, *i.e.* the synthetic images can only be used to obtain rough estimates of performance.

More accurate estimates of performance can be obtained by a more careful modelling of the errors in the actual range image as we did in Section B.3.

We also note that Section B.4 showed that there was a slight systematic component to the error. This implies that the errors obtained with real range data might be somewhat different from those obtained by using only the independent distribution of real errors.

Appendix C

B-rep model of widget

This appendix contains a B-rep model of the widget in Figure 4.44. The model was built by hand using the syntax described in Chapter 3.

Section C.1 contains the list of all the points in the model. Section C.2 lists the curves. The surfaces and the adjacency information are given in Section C.3.

C.1 Points

This section lists all the points in the B-rep model of the widget of Figure 4.44.

Figure C.1 shows the position of the listed points in the widget.

```
FEATURE : POINT
NAME : point_1
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION : (0.0 0.0 0.0)
CURVES : (edge_1 edge_5 edge_12)
PATCHES : (face_1 face_2 face_3 )
FEATURE : POINT

NAME : point_2
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION : (79.0 0.0 0.0)
CURVES : (edge_1 edge_2 edge_17)
PATCHES : (face_6 face_2 face_3 )


FEATURE : POINT
NAME : point_3
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION : (50.0 0.0 60.0)
CURVES : (edge_2 edge_3 edge_16)
PATCHES : (face_3 face_5 face_6 )

FEATURE : POINT
NAME : point_4
```

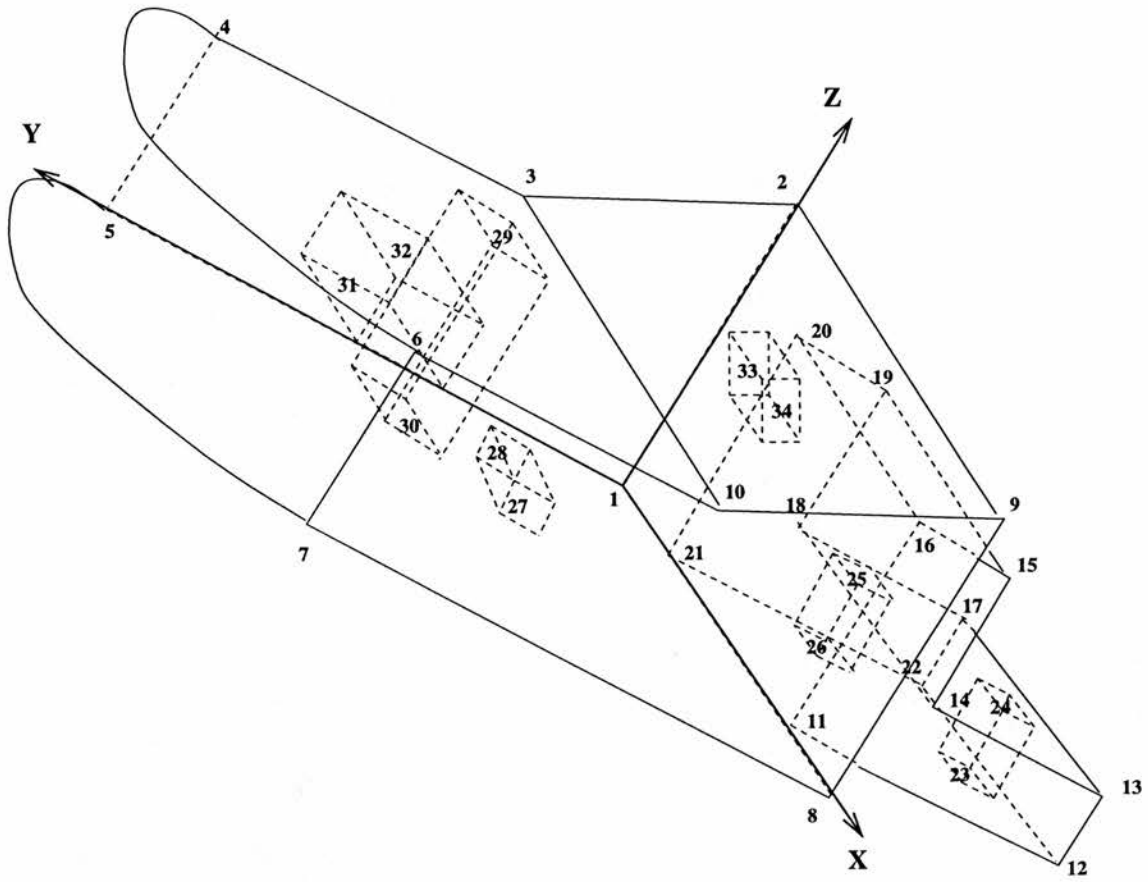


Figure C.1: Points in B-rep model of widget.

```
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION : (50.0 0.0 138.0)
CURVES : (edge_3 edge_4 edge_8)
PATCHES : (face_3 face_4 face_5 )

FEATURE : POINT
NAME : point_5
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION : (0.0 0.0 138.0)
CURVES : (edge_4 edge_5 edge_6)
PATCHES : (face_1 face_3 face_4 )

FEATURE : POINT
NAME : point_6
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (50.0 90.0 138.0)
CURVES : (edge_7 edge_8 edge_15)
PATCHES : (face_4 face_5 face_7)

FEATURE : POINT
NAME : point_7
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (0.0 90.0 138.0)
CURVES : (edge_6 edge_7 edge_9)
PATCHES : (face_1 face_4 face_7)

FEATURE : POINT
```

```
NAME : point_8
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (0.0 90.0 0.0)
CURVES : (edge_9 edge_10 edge_13)
PATCHES : (face_1 face_2 face_7)

FEATURE : POINT
NAME : point_9
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (79.0 90.0 0.0)
CURVES : (edge_9 edge_10 edge_13)
PATCHES : (face_1 face_6 face_7)

FEATURE : POINT
NAME : point_10
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (50.0 90.0 60.0)
CURVES : (edge_14 edge_15 edge_16)
PATCHES : (face_5 face_6 face_7)


FEATURE : POINT
NAME : point_11
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (0.0 75.0 0.0)
CURVES : (edge_9 edge_10 edge_13)
PATCHES : (face_1 face_6 face_7)

FEATURE : POINT
NAME : point_11
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (0.0 75.0 0.0)
CURVES : (edge_9 edge_10 edge_13)
PATCHES : (face_1 face_2 face_8 face_9)

FEATURE : POINT
NAME : point_12
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (0.0 75.0 -50.0)
CURVES : (edge_18 edge_19 edge_25)
PATCHES : (face_8 face_9 face_12)

FEATURE : POINT
NAME : point_13
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (20.0 75.0 -50.0)
CURVES : (edge_19 edge_20 edge_26)
PATCHES : (face_9 face_10 face_12)

FEATURE : POINT
NAME : point_14
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (20.0 75.0 -10.0)
CURVES : (edge_19 edge_20 edge_26)
PATCHES : (face_9 face_10 face_13)

FEATURE : POINT
NAME : point_15
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (60.0 75.0 -10.0)
CURVES : (edge_21 edge_22 edge_32)
PATCHES : (face_9 face_14 face_13)

FEATURE : POINT
NAME : point_16
```

```
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (60.0 75.0 0.0)
CURVES : (edge_22 edge_23 edge_33)
PATCHES : (face_2 face_9 face_14)

FEATURE : POINT
NAME : point_17
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (20.0 15.0 -50.0)
CURVES : (edge_26 edge_27 edge_28)
PATCHES : (face_10 face_11 face_12)

FEATURE : POINT
NAME : point_18
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (20.0 15.0 -10.0)
CURVES : (edge_26 edge_27 edge_28)
PATCHES : (face_10 face_11 face_12)

FEATURE : POINT
NAME : point_19
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (60.0 15.0 -10.0)
CURVES : (edge_29 edge_30 edge_32)
PATCHES : (face_11 face_13 face_14)

FEATURE : POINT
NAME : point_20
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (60.0 15.0 0.0)
CURVES : (edge_30 edge_31 edge_33)
PATCHES : (face_2 face_11 face_14)


FEATURE : POINT
NAME : point_21
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (0.0 15.0 0.0)
CURVES : (edge_12 edge_24 edge_31)
PATCHES : (face_1 face_2 face_8 face_11 )

FEATURE : POINT
NAME : point_22
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (0.0 15.0 -50.0)
CURVES : (edge_12 edge_24 edge_31)
PATCHES : (face_8 face_11 face_12)

FEATURE : POINT
NAME : point_23
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (0.0 60.0 -44.0)
CURVES : (edge_48 edge_49)
PATCHES : (face_8 face_22)

FEATURE : POINT
NAME : point_24
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (20.0 60.0 -44.0)
CURVES : (edge_47 edge_48)
PATCHES : (face_10 face_22)

FEATURE : POINT
NAME : point_25
POSITION
```

```
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (20.0 30.0 -30.0)
CURVES : (edge_50 edge_51)
PATCHES : (face_10 face_23)

FEATURE : POINT
NAME : point_26
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (0.0 30.0 -30.0)
CURVES : (edge_51 edge_52)
PATCHES : (face_8 face_23)

FEATURE : POINT
NAME : point_27
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (30.0 90.0 85.0)
CURVES : (edge_41 edge_42)
PATCHES : (face_19 face_7)

FEATURE : POINT
NAME : point_28
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (30.0 70.0 85.0)
CURVES : (edge_42 edge_43)
PATCHES : (face_18 face_19)

FEATURE : POINT
NAME : point_29
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (50.0 15.0 90.0)
CURVES : (edge_38 edge_39)
PATCHES : (face_5 face_17)

FEATURE : POINT
NAME : point_30
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (0.0 15.0 90.0)
CURVES : (edge_39 edge_40)
PATCHES : (face_1 face_17)

FEATURE : POINT
NAME : point_31
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (20.0 35.0 130.0)
CURVES : (edge_36 edge_37)
PATCHES : (face_15 face_16)

FEATURE : POINT
NAME : point_32
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (50.0 35.0 130.0)
CURVES : (edge_35 edge_36)
PATCHES : (face_5 face_16)

FEATURE : POINT
NAME : point_33
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
LOCATION :  (64.5 40.0 30.0)
CURVES : (edge_44 edge_45)
PATCHES : (face_6 face_21)

FEATURE : POINT
NAME : point_34
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
```

```
RST: (0.0 0.0 0.0)
LOCATION :  (46.5 40.0 21.3)
CURVES : (edge_45 edge_46)
PATCHES : (face_20 face_21)
```

C.2 Curves

This section lists all the curves in the B-rep model of the widget of Figure 4.44.

Figure C.2 shows the position of the listed curves in the widget.

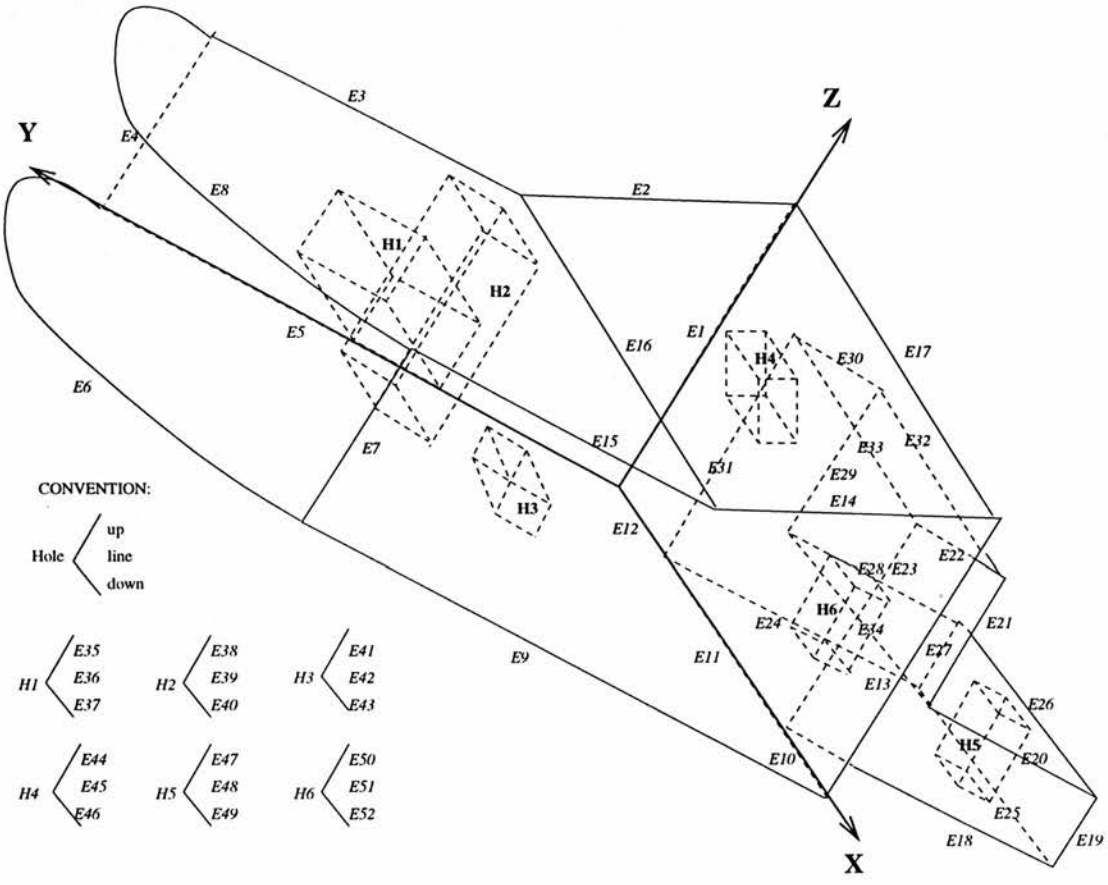


Figure C.2: Curves in B-rep model of widget.

```
FEATURE : LINE
NAME : edge_1
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (1.0 0.0 0.0)
BEGIN_POINT: point_1
END_POINT: point_2
PATCHES: (face_2 face_3)

FEATURE : LINE
NAME : edge_2
POSITION
TRANS: (79.0 0.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (-2.9 0.0 6.0)
BEGIN_POINT: point_2
END_POINT: point_3
```

```
PATCHES: (face_3 face_6)

FEATURE : LINE
NAME : edge_3
POSITION
TRANS: (50.0 0.0 60.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_3
END_POINT: point_4
PATCHES: (face_3 face_5)

FEATURE : LINE
NAME : edge_4
POSITION
TRANS: (50.0 0.0 138.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (-1.0 0.0 0.0)
BEGIN_POINT: point_4
END_POINT: point_5
PATCHES: (face_3 face_4)

FEATURE : LINE
NAME : edge_5
POSITION
TRANS: (0.0 0.0 138.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (0.0 0.0 -1.0)
BEGIN_POINT: point_5
END_POINT: point_1
PATCHES: (face_1 face_3)

FEATURE : ELLIPSE
NAME : edge_6
POSITION
TRANS: (0.0 45.0 138.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (0.0 0.0 1.0)
BEGIN_POINT: point_7
END_POINT: point_5
PATCHES: (face_1 face_4)
RADIUS_X: 40.0
RADIUS_Y: 45.0

FEATURE : LINE
NAME : edge_7
POSITION
TRANS: (50.0 90.0 138.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (-1.0 0.0 0.0)
BEGIN_POINT: point_6
END_POINT: point_7
PATCHES: (face_4 face_7)

FEATURE : ELLIPSE
NAME : edge_8
POSITION
TRANS: (50.0 45.0 138.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (-1.0 0.0 0.0)
BEGIN_POINT: point_4
END_POINT: point_6
PATCHES: (face_4 face_5)
RADIUS_X: 40.0
RADIUS_Y: 45.0

FEATURE : LINE
NAME : edge_9
POSITION
TRANS: (0.0 90.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_8
END_POINT: point_7
PATCHES: (face_1 face_7)

FEATURE : LINE
NAME : edge_10
POSITION
TRANS: (0.0 90.0 0.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (0.0 -1.0 0.0)
BEGIN_POINT: point_8
END_POINT: point_11
PATCHES: (face_2 face_1)
```

```
FEATURE : LINE
NAME : edge_11
POSITION
TRANS: (0.0 75.0 0.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (0.0 -1.0 0.0)
BEGIN_POINT: point_11
END_POINT: point_21
PATCHES: (face_1 face_8)

FEATURE : LINE
NAME : edge_12
POSITION
TRANS: (0.0 15.0 0.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (0.0 -1.0 0.0)
BEGIN_POINT: point_21
END_POINT: point_1
PATCHES: (face_2 face_1)

FEATURE : LINE
NAME : edge_13
POSITION
TRANS: (0.0 90.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (1.0 0.0 0.0)
BEGIN_POINT: point_8
END_POINT: point_9
PATCHES: (face_2 face_7)

FEATURE : LINE
NAME : edge_14
POSITION
TRANS: (79.0 90.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (-2.9 0.0 6.0)
BEGIN_POINT: point_9
END_POINT: point_10
PATCHES: (face_7 face_11)

FEATURE : LINE
NAME : edge_15
POSITION
TRANS: (50.0 90.0 60.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_10
END_POINT: point_6
PATCHES: (face_5 face_7)

FEATURE : LINE
NAME : edge_16
POSITION
TRANS: (50.0 90.0 60.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (0.0 -1.0 0.0)
BEGIN_POINT: point_10
END_POINT: point_3
PATCHES: (face_5 face_5)

FEATURE : LINE
NAME : edge_17
POSITION
TRANS: (79.0 0.0 0.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (0.0 1.0 0.0)
BEGIN_POINT: point_2
END_POINT: point_9
PATCHES: (face_2 face_6)

FEATURE : LINE
NAME : edge_18
POSITION
TRANS: (0.0 75.0 0.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (0.0 0.0 -1.0)
BEGIN_POINT: point_11
END_POINT: point_12
PATCHES: (face_8 face_9)

FEATURE : LINE
NAME : edge_19
POSITION
TRANS: (0.0 75.0 -50.0)
ROTATION
```

```
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (1.0 0.0 0.0)
BEGIN_POINT: point_12
END_POINT: point_13
PATCHES: (face_9 face_12)

FEATURE : LINE
NAME : edge_20
POSITION
TRANS: (20.0 75.0 -50.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_13
END_POINT: point_14
PATCHES: (face_9 face_10)

FEATURE : LINE
NAME : edge_21
POSITION
TRANS: (20.0 75.0 -10.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (1.0 0.0 0.0)
BEGIN_POINT: point_14
END_POINT: point_15
PATCHES: (face_9 face_13)


FEATURE : LINE
NAME : edge_22
POSITION
TRANS: (60.0 75.0 -10.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_15
END_POINT: point_16
PATCHES: (face_9 face_14)

FEATURE : LINE
NAME : edge_23
POSITION
TRANS: (0.0 75.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (1.0 0.0 0.0)
BEGIN_POINT: point_11
END_POINT: point_16
PATCHES: (face_9 face_2)

FEATURE : LINE
NAME : edge_24
POSITION
TRANS: (0.0 15.0 -50.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_22
END_POINT: point_21
PATCHES: (face_8 face_11)

FEATURE : LINE
NAME : edge_25
POSITION
TRANS: (0.0 15.0 -50.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (0.0 1.0 0.0)
BEGIN_POINT: point_22
END_POINT: point_12
PATCHES: (face_8 face_12)

FEATURE : LINE
NAME : edge_26
POSITION
TRANS: (20.0 75.0 -50.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (0.0 -1.0 0.0)
BEGIN_POINT: point_13
END_POINT: point_17
PATCHES: (face_12 face_10)

FEATURE : LINE
NAME : edge_27
POSITION
TRANS: (20.0 15.0 -50.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (-1.0 0.0 0.0)
BEGIN_POINT: point_17
END_POINT: point_22
PATCHES: (face_11 face_12)
```

```
FEATURE : LINE
NAME : edge_28
POSITION
TRANS: (20.0 15.0 -50.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_17
END_POINT: point_18
PATCHES: (face_10 face_11)

FEATURE : LINE
NAME : edge_29
POSITION
TRANS: (20.0 15.0 -10.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (1.0 0.0 0.0)
BEGIN_POINT: point_18
END_POINT: point_19
PATCHES: (face_11 face_13)

FEATURE : LINE
NAME : edge_30
POSITION
TRANS: (60.0 15.0 -10.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_19
END_POINT: point_20
PATCHES: (face_11 face_14)

FEATURE : LINE
NAME : edge_31
POSITION
TRANS: (60.0 15.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (-1.0 0.0 0.0)
BEGIN_POINT: point_20
END_POINT: point_21
PATCHES: (face_2 face_11)


FEATURE : LINE
NAME : edge_32
POSITION
TRANS: (60.0 75.0 -10.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (0.0 -1.0 0.0)
BEGIN_POINT: point_15
END_POINT: point_19
PATCHES: (face_13 face_14)

FEATURE : LINE
NAME : edge_33
POSITION
TRANS: (60.0 75.0 0.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (0.0 -1.0 0.0)
BEGIN_POINT: point_16
END_POINT: point_20
PATCHES: (face_2 face_14)

FEATURE : LINE
NAME : edge_34
POSITION
TRANS: (20.0 15.0 -10.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (0.0 1.0 0.0)
BEGIN_POINT: point_18
END_POINT: point_14
PATCHES: (face_11 face_13)

FEATURE : CIRCLE
NAME : edge_35
POSITION
TRANS: (50.0 45.0 130.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (0.0 0.0 1.0)
BEGIN_POINT: point_32
END_POINT: point_32
PATCHES: (face_5 face_16)
RADIUS: 10.0

FEATURE : LINE
NAME : edge_36
POSITION
TRANS: (50.0 35.0 130.0)
```

```
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (-1.0 0.0 0.0)
BEGIN_POINT: point_32
END_POINT: point_31
PATCHES: (face_16)

FEATURE : CIRCLE
NAME : edge_37
POSITION
TRANS: (20.0 45.0 130.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (0.0 0.0 1.0)
BEGIN_POINT: point_31
END_POINT: point_31
PATCHES: (face_15 face_16)
RADIUS: 10.0

FEATURE : CIRCLE
NAME : edge_38
POSITION
TRANS: (50.0 20.0 90.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (0.0 0.0 1.0)
BEGIN_POINT: point_29
END_POINT: point_29
PATCHES: (face_5 face_17)
RADIUS: 5.0

FEATURE : LINE
NAME : edge_39
POSITION
TRANS: (50.0 15.0 90.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (-1.0 0.0 0.0)
BEGIN_POINT: point_29
END_POINT: point_30
PATCHES: (face_17)

FEATURE : CIRCLE
NAME : edge_40
POSITION
TRANS: (0.0 20.0 90.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (0.0 0.0 1.0)
BEGIN_POINT: point_30
END_POINT: point_30
PATCHES: (face_1 face_17)
RADIUS: 5.0

FEATURE : CIRCLE
NAME : edge_41
POSITION
TRANS: (30.0 90.0 90.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 0.0 1.0)
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
BEGIN_POINT: point_27
END_POINT: point_27
PATCHES: (face_7 face_19)
RADIUS: 5.0


FEATURE : LINE
NAME : edge_42
POSITION
TRANS: (30.0 90.0 85.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (0.0 -1.0 0.0)
BEGIN_POINT: point_27
END_POINT: point_28
PATCHES: (face_19)

FEATURE : CIRCLE
NAME : edge_43
POSITION
TRANS: (30.0 70.0 90.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 0.0 1.0)
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
BEGIN_POINT: point_28
END_POINT: point_28
PATCHES: (face_18 face_19)
RADIUS: 5.0

FEATURE : CIRCLE
NAME : edge_44
```

```
POSITION
TRANS: (64.5 45.0 30.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (-2.9 0.0 6.0)
BEGIN_POINT: point_33
END_POINT: point_33
PATCHES: (face_6 face_21)
RADIUS: 4.0

FEATURE : LINE
NAME : edge_45
POSITION
TRANS: (46.5 40.0 21.3)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (6.0 0.0 2.9)
BEGIN_POINT: point_34
END_POINT: point_33
PATCHES: (face_21)

FEATURE : CIRCLE
NAME : edge_46
POSITION
TRANS: (46.5 45.0 21.3)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (2.9 0.0 -6.0)
BEGIN_POINT: point_34
END_POINT: point_34
PATCHES: (face_20 face_21)
RADIUS: 4.0

FEATURE : CIRCLE
NAME : edge_47
POSITION
TRANS: (20.0 60.0 -40.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (0.0 0.0 1.0)
BEGIN_POINT: point_24
END_POINT: point_24
PATCHES: (face_10 face_22)
RADIUS: 4.0

FEATURE : LINE
NAME : edge_48
POSITION
TRANS: (20.0 60.0 -44.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (-1.0 0.0 0.0)
BEGIN_POINT: point_24
END_POINT: point_23
PATCHES: (face_22)

FEATURE : CIRCLE
NAME : edge_49
POSITION
TRANS: (0.0 60.0 -40.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (0.0 0.0 1.0)
BEGIN_POINT: point_23
END_POINT: point_23
PATCHES: (face_8 face_22)
RADIUS: 4.0

FEATURE : CIRCLE
NAME : edge_50
POSITION
TRANS: (20.0 30.0 -25.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (0.0 0.0 1.0)
BEGIN_POINT: point_25
END_POINT: point_25
PATCHES: (face_10 face_23)
RADIUS: 5.0

FEATURE : LINE
NAME : edge_51
POSITION
TRANS: (20.0 30.0 -25.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (-1.0 0.0 0.0)
BEGIN_POINT: point_25
END_POINT: point_26
PATCHES: (face_23)

FEATURE : CIRCLE
```

```
NAME : edge_52
POSITION
TRANS: (0.0 30.0 -25.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (0.0 0.0 1.0)
BEGIN_POINT: point_26
END_POINT: point_26
PATCHES: (face_8 face_23)
RADIUS: 5.0


FEATURE : POLYCURVE
NAME : bound_1
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (-1.0 0.0 0.0)
BEGIN_POINT: point_1
END_POINT: point_1
PATCHES: (face_1)
CURVES: (edge_5 edge_6 edge_9 edge_10 edge_11 edge_12)

FEATURE : POLYCURVE
NAME : bound_1_2
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (-1.0 0.0 0.0)
BEGIN_POINT: point_30
END_POINT: point_30
PATCHES: (face_1)
CURVES: (edge_40)

FEATURE : POLYCURVE
NAME : bound_2
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_1
END_POINT: point_1
PATCHES: (face_2)
CURVES: (edge_1 edge_17 edge_13 edge_10 edge_23 edge_33 edge_31 edge_12)


FEATURE : POLYCURVE
NAME : bound_3
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0)  (0.0 0.0 1.0)
VEC_TO_VEC: (0.0 1.0 0.0)  (1.0 0.0 0.0)
BEGIN_POINT: point_1
END_POINT: point_1
PATCHES: (face_3)
CURVES: (edge_1 edge_2 edge_3 edge_4 edge_5)

FEATURE : POLYCURVE
NAME : bound_4
POSITION
TRANS: (0.0 45.0 138.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_4
END_POINT: point_4
PATCHES: (face_4)
CURVES: (edge_4 edge_6 edge_7 edge_8)

FEATURE : POLYCURVE
NAME : bound_5
POSITION
TRANS: (50.0 0.0 60.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (1.0 0.0 0.0)
BEGIN_POINT: point_4
END_POINT: point_4
PATCHES: (face_5)
CURVES: (edge_8 edge_15 edge_16 edge_3)

FEATURE : POLYCURVE
NAME : bound_5_2
POSITION
TRANS: (50.0 0.0 60.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (1.0 0.0 0.0)
BEGIN_POINT: point_32
END_POINT: point_32
```

```
PATCHES: (face_5)
CURVES: (edge_35)

FEATURE : POLYCURVE
NAME : bound_5_3
POSITION
TRANS: (50.0 0.0 60.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (1.0 0.0 0.0)
BEGIN_POINT: point_29
END_POINT: point_29
PATCHES: (face_5)
CURVES: (edge_38)

FEATURE : POLYCURVE
NAME : bound_6
POSITION
TRANS: (79.0 0.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0)  (-2.9 0.0 6.0)
BEGIN_POINT: point_3
END_POINT: point_3
PATCHES: (face_6)
CURVES: (edge_16 edge_14 edge_17 edge_2)

FEATURE : POLYCURVE
NAME : bound_6_2
POSITION
TRANS: (79.0 0.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0)  (-2.9 0.0 6.0)
BEGIN_POINT: point_33
END_POINT: point_33
PATCHES: (face_6)
CURVES: (edge_44)

FEATURE : POLYCURVE
NAME : bound_7
POSITION
TRANS: (0.0 90.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (0.0 1.0 0.0)
BEGIN_POINT: point_7
END_POINT: point_7
PATCHES: (face_7)
CURVES: (edge_7 edge_9 edge_13 edge_14 edge_15)


FEATURE : POLYCURVE
NAME : bound_7_2
POSITION
TRANS: (0.0 90.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (0.0 1.0 0.0)
BEGIN_POINT: point_27
END_POINT: point_27
PATCHES: (face_7)
CURVES: (edge_41)

FEATURE : POLYCURVE
NAME : bound_8
POSITION
TRANS: (0.0 15.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (-1.0 0.0 0.0)
BEGIN_POINT: point_11
END_POINT: point_11
PATCHES: (face_8)
CURVES: (edge_18 edge_25 edge_24 edge_11)

FEATURE : POLYCURVE
NAME : bound_8_2
POSITION
TRANS: (0.0 15.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (-1.0 0.0 0.0)
BEGIN_POINT: point_23
END_POINT: point_23
PATCHES: (face_8)
CURVES: (edge_49)

FEATURE : POLYCURVE
NAME : bound_8_3
POSITION
TRANS: (0.0 15.0 0.0)
```

```
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (-1.0 0.0 0.0)
BEGIN_POINT: point_26
END_POINT: point_26
PATCHES: (face_8)
CURVES: (edge_52)

FEATURE : POLYCURVE
NAME : bound_9
POSITION
TRANS: (0.0 75.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (0.0 1.0 0.0)
BEGIN_POINT: point_11
END_POINT: point_11
PATCHES: (face_9)
CURVES: (edge_18 edge_19 edge_20 edge_21 edge_22 edge_23)

FEATURE : POLYCURVE
NAME : bound_10
POSITION
TRANS: (20.0 15.0 -10.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (1.0 0.0 0.0)
BEGIN_POINT: point_13
END_POINT: point_13
PATCHES: (face_10)
CURVES: (edge_20 edge_34 edge_28 edge_26)

FEATURE : POLYCURVE
NAME : bound_10_2
POSITION
TRANS: (20.0 15.0 -10.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (1.0 0.0 0.0)
BEGIN_POINT: point_24
END_POINT: point_24
PATCHES: (face_10)
CURVES: (edge_47)

FEATURE : POLYCURVE
NAME : bound_10_3
POSITION
TRANS: (20.0 15.0 -10.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (1.0 0.0 0.0)
BEGIN_POINT: point_25
END_POINT: point_25
PATCHES: (face_10)
CURVES: (edge_50)

FEATURE : POLYCURVE
NAME : bound_11
POSITION
TRANS: (0.0 15.0 0.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0)  (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (0.0 -1.0 0.0)
BEGIN_POINT: point_11
END_POINT: point_11
PATCHES: (face_11)
CURVES: (edge_27 edge_28 edge_29 edge_30 edge_31 edge_24)

FEATURE : POLYCURVE
NAME : bound_12
POSITION
TRANS: (0.0 15.0 -50.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_12
END_POINT: point_12
PATCHES: (face_12)
CURVES: (edge_19 edge_26 edge_27 edge_25)


FEATURE : POLYCURVE
NAME : bound_13
POSITION
TRANS: (20.0 15.0 -10.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_14
END_POINT: point_14
PATCHES: (face_13)
CURVES: (edge_21 edge_32 edge_29 edge_34)

FEATURE : POLYCURVE
```

```
NAME : bound_14
POSITION
TRANS: (60.0 15.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (1.0 0.0 0.0)
BEGIN_POINT: point_15
END_POINT: point_15
PATCHES: (face_14)
CURVES: (edge_22 edge_33 edge_30 edge_32)

FEATURE : POLYCURVE
NAME : bound_15
POSITION
TRANS: (20.0 45.0 130.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (0.0 0.0 1.0)
BEGIN_POINT: point_31
END_POINT: point_31
PATCHES: (face_15)
CURVES: (edge_37)

FEATURE : POLYCURVE
NAME : bound_16
POSITION
TRANS: (20.0 45.0 130.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_31
END_POINT: point_32
PATCHES: (face_11)
CURVES: (edge_37 edge_36 edge_35)

FEATURE : POLYCURVE
NAME : bound_17
POSITION
TRANS: (0.0 20.0 90.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_29
END_POINT: point_30
PATCHES: (face_11)
CURVES: (edge_38 edge_39 edge_40)

FEATURE : POLYCURVE
NAME : bound_18
POSITION
TRANS: (30.0 70.0 90.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 0.0 1.0)
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
BEGIN_POINT: point_28
END_POINT: point_28
PATCHES: (face_18)
CURVES: (edge_43)

FEATURE : POLYCURVE
NAME : bound_19
POSITION
TRANS: (30.0 70.0 90.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (0.0 0.0 1.0)
BEGIN_POINT: point_27
END_POINT: point_28
PATCHES: (face_19)
CURVES: (edge_41 edge_42 edge_43)

FEATURE : POLYCURVE
NAME : bound_20
POSITION
TRANS: (48.4 45.0 19.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (2.9 0.0 -6.0)
BEGIN_POINT: point_34
END_POINT: point_34
PATCHES: (face_20)
CURVES: (edge_46)

FEATURE : POLYCURVE
NAME : bound_21
POSITION
TRANS: (46.5 45.0 21.3)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (6.0 0.0 2.9)
BEGIN_POINT: point_33
END_POINT: point_34
PATCHES: (face_21)
CURVES: (edge_44 edge_45 edge_46)
```

```
FEATURE : POLYCURVE
NAME : bound_22
POSITION
TRANS: (0.0 60.0 -40.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_23
END_POINT: point_24
PATCHES: (face_22)
CURVES: (edge_49 edge_48 edge_47)

FEATURE : POLYCURVE
NAME : bound_23
POSITION
TRANS: (0.0 30.0 -25.0)
ROTATION
RST: (0.0 0.0 0.0)
BEGIN_POINT: point_25
END_POINT: point_26
PATCHES: (face_23)
CURVES: (edge_50 edge_51 edge_52)
```

C.3 Surfaces and adjacency information

This section lists all the surfaces in the B-rep model of the widget of Figure 4.44. The adjacency between the surfaces is also described after the list of surfaces.

Figure C.3 shows the position of the listed surfaces in the widget.

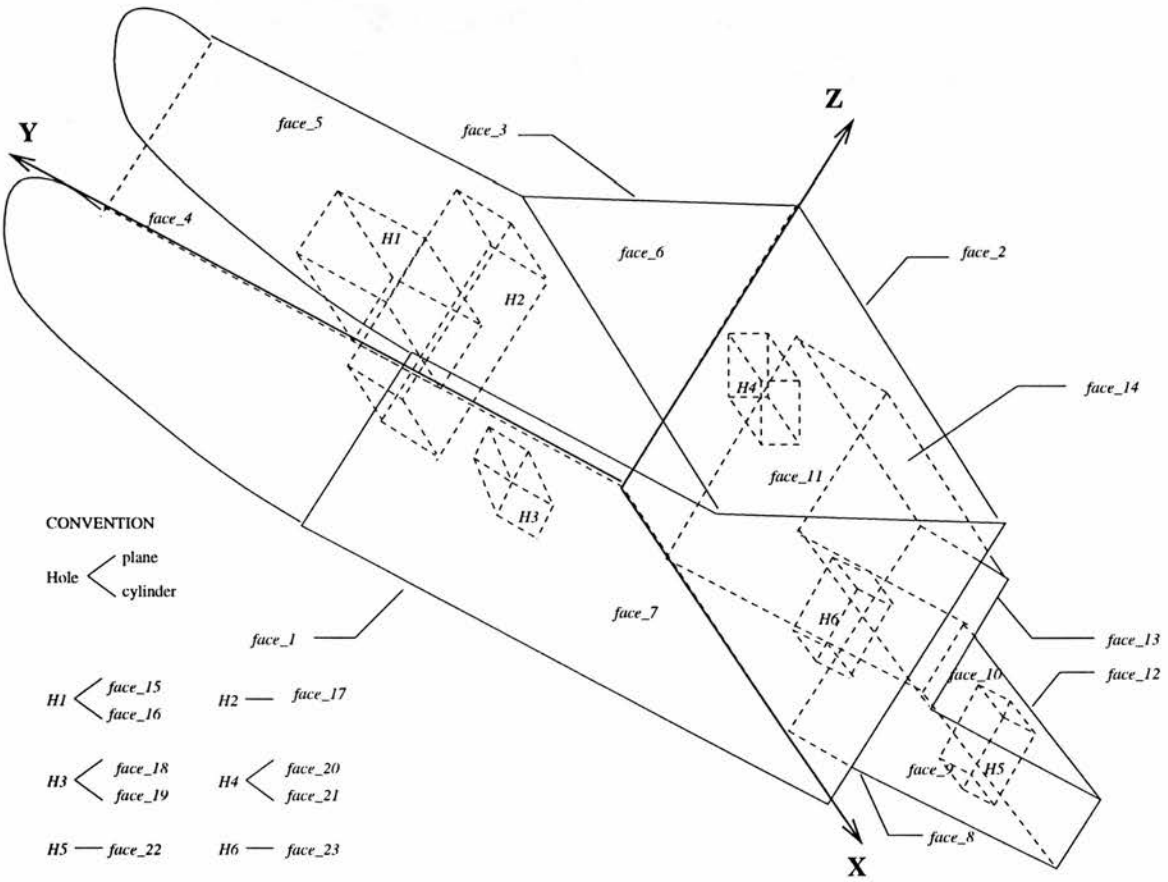


Figure C.3: Surfaces in B-rep model of widget.

```
FEATURE : PLANE
```

```
NAME : face_1
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0)  (1.0 0.0 0.0)
INSIDE_POINT: (-40.0 45.0 0.0)
LOOPS
CURVES: (bound_1 bound_1_2)
NORMAL: (0.0 0.0 -1.0)

FEATURE : PLANE
NAME : face_2
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
RST: (0.0 0.0 0.0)
INSIDE_POINT: (40.0 8.0 0.0)
LOOPS
CURVES: (bound_2)
NORMAL: (0.0 0.0 -1.0)

FEATURE : PLANE
NAME : face_3
POSITION
TRANS: (0.0 0.0 0.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0)  (0.0 0.0 1.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (0.0 -1.0 0.0)
INSIDE_POINT: (60.0 25.0 0.0)
LOOPS
CURVES: (bound_3)
NORMAL: (0.0 0.0 -1.0)

FEATURE : CYLINDER
NAME : face_4
POSITION
TRANS: (0.0 45.0 138.0)
ROTATION
RST: (0.0 0.0 0.0)
INSIDE_POINT: (25.0 0.0 40.0)
LOOPS
CURVES: (bound_4)
RADIUS_Y: 45.0
RADIUS_Z: 40.0
TETA_BEGIN: -1.5708
TETA_END: 1.5708
X_BEGIN: 0.0
X_END: 50.0


FEATURE : PLANE
NAME : face_5
POSITION
TRANS: (50.0 0.0 60.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (1.0 0.0 0.0)
INSIDE_POINT: (40.0 45.0 0.0)
LOOPS
CURVES: (bound_5 bound_5_2 bound_5_3)
NORMAL: (0.0 0.0 -1.0)

FEATURE : PLANE
NAME : face_6
POSITION
TRANS: (79.0 0.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0)  (-2.9 0.0 6.0)
INSIDE_POINT: (20.0 30.0 0.0)
LOOPS
CURVES: (bound_6 bound_6_2)
NORMAL: (0.0 0.0 -1.0)

FEATURE : PLANE
NAME : face_7
POSITION
TRANS: (0.0 90.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (0.0 1.0 0.0)
INSIDE_POINT: (-25.0 20.0 0.0)
LOOPS
CURVES: (bound_7)
NORMAL: (0.0 0.0 -1.0)

FEATURE : PLANE
NAME : face_8
POSITION
TRANS: (0.0 15.0 0.0)
ROTATION
```

```
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (-1.0 0.0 0.0)
INSIDE_POINT: (20.0 30.0 0.0)
LOOPS
CURVES: (bound_8 bound_8_2 bound_8_3)
NORMAL: (0.0 0.0 -1.0)

FEATURE : PLANE
NAME : face_9
POSITION
TRANS: (0.0 75.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (0.0 1.0 0.0)
INSIDE_POINT: (40.0 10.0 0.0)
LOOPS
CURVES: (bound_9)
NORMAL: (0.0 0.0 -1.0)

FEATURE : PLANE
NAME : face_10
POSITION
TRANS: (20.0 15.0 -10.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (1.0 0.0 0.0)
INSIDE_POINT: (-30.0 15 0.0)
LOOPS
CURVES: (bound_10 bound_10_2 bound_10_3)
NORMAL: (0.0 0.0 -1.0)

FEATURE : PLANE
NAME : face_11
POSITION
TRANS: (0.0 15.0 0.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0)  (1.0 0.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (0.0 -1.0 0.0)
INSIDE_POINT: (20.0 5.0 0.0)
LOOPS
CURVES: (bound_11)
NORMAL: (0.0 0.0 -1.0)

FEATURE : PLANE
NAME : face_12
POSITION
TRANS: (0.0 15.0 -50.0)
ROTATION
RST: (0.0 0.0 0.0)
INSIDE_POINT: (10.0 30.0 0.0)
LOOPS
CURVES: (bound_12)
NORMAL: (0.0 0.0 -1.0)

FEATURE : PLANE
NAME : face_13
POSITION
TRANS: (20.0 15.0 -10.0)
ROTATION
RST: (0.0 0.0 0.0)
INSIDE_POINT: (20.0 30.0 0.0)
LOOPS
CURVES: (bound_13)
NORMAL: (0.0 0.0 -1.0)

FEATURE : PLANE
NAME : face_14
POSITION
TRANS: (60.0 15.0 0.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0)  (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 -1.0)  (1.0 0.0 0.0)
INSIDE_POINT: (-5.0 30.0 0.0)
LOOPS
CURVES: (bound_14)
NORMAL: (0.0 0.0 -1.0)

FEATURE : PLANE
NAME : face_15
POSITION
TRANS: (20.0 45.0 130.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (0.0 0.0 1.0)
INSIDE_POINT: (0.0 0.0 0.0)
LOOPS
CURVES: (bound_15)
NORMAL: (0.0 0.0 -1.0)

FEATURE : CYLINDER
NAME : face_16
POSITION
```

```
TRANS: (20.0 45.0 130.0)
ROTATION
RST: (0.0 0.0 0.0)
INSIDE_POINT: (20.0 0.0 10.0)
LOOPS
CURVES: (bound_16)
RADIUS_Y: -10.0
RADIUS_Z: -10.0
TETA_BEGIN: 0.0
TETA_END: 6.2831853
X_BEGIN: 0.0
X_END: 30.0


FEATURE : CYLINDER
NAME : face_17
POSITION
TRANS: (0.0 20.0 90.0)
ROTATION
RST: (0.0 0.0 0.0)
INSIDE_POINT: (20.0 0.0 -5.0)
LOOPS
CURVES: (bound_17)
RADIUS_Y: -5.0
RADIUS_Z: -5.0
TETA_BEGIN: 0.0
TETA_END: 6.2831853
X_BEGIN: 0.0
X_END: 50.0

FEATURE : PLANE
NAME : face_18
POSITION
TRANS: (30.0 70.0 90.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 0.0 1.0)
VEC_TO_VEC: (1.0 0.0 0.0) (1.0 0.0 0.0)
INSIDE_POINT: (0.0 0.0 0.0)
LOOPS
CURVES: (bound_18)
NORMAL: (0.0 0.0 -1.0)

FEATURE : CYLINDER
NAME : face_19
POSITION
TRANS: (30.0 70.0 90.0)
ROTATION
VEC_TO_VEC: (1.0 0.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (0.0 0.0 1.0) (0.0 0.0 1.0)
INSIDE_POINT: (10.0 0.0 5.0)
LOOPS
CURVES: (bound_19)
RADIUS_Y: -5.0
RADIUS_Z: -5.0
TETA_BEGIN: 0.0
TETA_END: 6.2831853
X_BEGIN: 0.0
X_END: 20.0

FEATURE : PLANE
NAME : face_20
POSITION
TRANS: (48.4 45.0 19.0)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (2.9 0.0 -6.0)
INSIDE_POINT: (0.0 0.0 0.0)
LOOPS
CURVES: (bound_20)
NORMAL: (0.0 0.0 -1.0)

FEATURE : CYLINDER
NAME : face_21
POSITION
TRANS: (46.5 45.0 21.3)
ROTATION
VEC_TO_VEC: (0.0 1.0 0.0) (0.0 1.0 0.0)
VEC_TO_VEC: (1.0 0.0 0.0) (6.0 0.0 2.9)
INSIDE_POINT: (10.0 0.0 5.0)
LOOPS
CURVES: (bound_21)
RADIUS_Y: -5.0
RADIUS_Z: -5.0
TETA_BEGIN: 0.0
TETA_END: 6.2831853
X_BEGIN: 0.0
X_END: 20.0

FEATURE : CYLINDER
NAME : face_22
POSITION
TRANS: (0.0 60.0 -40.0)
```

```
ROTATION
RST: (0.0 0.0 0.0)
INSIDE_POINT: (10.0 0.0 4.0)
LOOPS
CURVES: (bound_22)
RADIUS_Y: -4.0
RADIUS_Z: -4.0
TETA_BEGIN: 0.0
TETA_END: 6.2831853
X_BEGIN: 0.0
X_END: 20.0

FEATURE : CYLINDER
NAME : face_23
POSITION
TRANS: (0.0 30.0 -25.0)
ROTATION
RST: (0.0 0.0 0.0)
INSIDE_POINT: (10.0 0.0 -5.0)
LOOPS
CURVES: (bound_23)
RADIUS_Y: -5.0
RADIUS_Z: -5.0
TETA_BEGIN: 0.0
TETA_END: 6.2831853
X_BEGIN: 0.0
X_END: 20.0


ADJACENCY
NAME : face_1
PATCHES : (face_8 face_2 face_3 face_4 face_7 face_17)

ADJACENCY
NAME : face_2
PATCHES : (face_9 face_14 face_11 face_6 face_1)

ADJACENCY
NAME : face_3
PATCHES : (face_1 face_4 face_5 face_6 face_2)

ADJACENCY
NAME : face_4
PATCHES : (face_3 face_7 face_5 face_1)

ADJACENCY
NAME : face_5
PATCHES : (face_3 face_6 face_4 face_7 face_17 face_16)

ADJACENCY
NAME : face_6
PATCHES : (face_3 face_2 face_5 face_7 face_21)

ADJACENCY
NAME : face_7
PATCHES : (face_4 face_5 face_6 face_2 face_1)

ADJACENCY
NAME : face_8
PATCHES : (face_1 face_9 face_11 face_12 face_22 face_23)

ADJACENCY
NAME : face_9
PATCHES : (face_13 face_14 face_2 face_10 face_12 face_8)

ADJACENCY
NAME : face_10
PATCHES : (face_13 face_9 face_11 face_12 face_22 face_23)

ADJACENCY
NAME : face_11
PATCHES : (face_2 face_14 face_13 face_10 face_12 face_8)

ADJACENCY
NAME : face_12
PATCHES : (face_9 face_11 face_2 face_10 face_8)

ADJACENCY
NAME : face_13
PATCHES : (face_9 face_11 face_14 face_10)

ADJACENCY
NAME : face_14
PATCHES : (face_2 face_11 face_9 face_13)

ADJACENCY
NAME : face_15
PATCHES : (face_16)

ADJACENCY
NAME : face_16
PATCHES : (face_15 face_5)
```

```
ADJACENCY
NAME : face_17
PATCHES : (face_1 face_5)

ADJACENCY
NAME : face_18
PATCHES : (face_19)

ADJACENCY
NAME : face_19
PATCHES : (face_18 face_7)

ADJACENCY
NAME : face_20
PATCHES : (face_21)

ADJACENCY
NAME : face_21
PATCHES : (face_6 face_20)

ADJACENCY
NAME : face_22
PATCHES : (face_10 face_8)

ADJACENCY
NAME : face_23
PATCHES : (face_10 face_8)
```

Appendix D

DT model of widget

This appendix contains a DT model of the widget in Figure 4.44. The model was built by hand using the syntax described in Chapter 3.

Section D.1 contains a list of tolerances associated with the features described in Appendix C.

D.1 Tolerances

```
DT_NODE
FEATURE: PLANE
NAME: face_1
DRF
TYPE : THREE_FACES
FACE : (face_1)
FACE : (face_2)
FACE : (face_3)
ED_SIZE
MAT_COND: NONE
TOLERANCE: NONE
ED_FORM
MAT_COND: M
TOLERANCE : CONSTANT
RADIUS : 1.0
TYPE : FLATNESS
ED_POS
MAT_COND: NONE
TOLERANCE: NONE

DT_NODE
FEATURE: PLANE
NAME: face_2
DRF
TYPE : THREE_FACES
FACE : (face_1)
FACE : (face_2)
FACE : (face_3)
ED_SIZE
MAT_COND: NONE
TOLERANCE: NONE
ED_FORM
MAT_COND: M
TOLERANCE : CONSTANT
RADIUS : 1.0
TYPE : FLATNESS
ED_POS
MAT_COND: NONE
TOLERANCE: NONE

DT_NODE
FEATURE: PLANE
NAME: face_3
DRF
TYPE : THREE_FACES
FACE : (face_1)
```

```
FACE : (face_2)
FACE : (face_3)
ED_SIZE
MAT_COND: NONE
TOLERANCE: NONE
ED_FORM
MAT_COND: M
TOLERANCE : CONSTANT
RADIUS : 1.0
TYPE : FLATNESS
ED_POS
MAT_COND: NONE
TOLERANCE: NONE

DT_NODE
FEATURE: PLANE
NAME: face_7
DRF
TYPE : ONE_FACE
FACE : (face_3)
ED_SIZE
MAT_COND: NONE
TOLERANCE: NONE
ED_FORM
MAT_COND: M
TOLERANCE : CONSTANT
RADIUS: 1.0
TYPE: PARALLEL
ED_POS
MAT_COND: NONE
TOLERANCE: NONE

DT_NODE
FEATURE: PLANE
NAME: face_9
DRF
TYPE : ONE_FACE
FACE : (face_3)
ED_SIZE
MAT_COND: NONE
TOLERANCE: NONE
ED_FORM
MAT_COND: M
TOLERANCE : CONSTANT
RADIUS: 1.0
TYPE: PARALLEL
ED_POS
MAT_COND: NONE
TOLERANCE: NONE

DT_NODE
FEATURE: PLANE
NAME: face_11
DRF
TYPE : ONE_FACE
FACE : (face_3)
ED_SIZE
MAT_COND: NONE
TOLERANCE: NONE
ED_FORM
MAT_COND: M
TOLERANCE : CONSTANT
RADIUS: 1.0
TYPE: PARALLEL
ED_POS
MAT_COND: NONE
TOLERANCE: NONE

DT_NODE
FEATURE: PLANE
NAME: face_3
DRF
TYPE : ONE_FACE
FACE : (face_1)
ED_SIZE
MAT_COND: NONE
TOLERANCE: NONE
ED_FORM
MAT_COND: M
TOLERANCE : CONSTANT
RADIUS: 1.0
TYPE: PERPENDICULAR
ED_POS
MAT_COND: NONE
TOLERANCE: NONE

DT_NODE
FEATURE: PLANE
NAME: face_2
DRF
TYPE : ONE_FACE
FACE : (face_1)
```

```
ED_SIZE
MAT_COND: NONE
TOLERANCE: NONE
ED_FORM
MAT_COND: M
TOLERANCE : CONSTANT
RADIUS: 1.0
TYPE: PERPENDICULAR
ED_POS
MAT_COND: NONE
TOLERANCE: NONE

DT_NODE
FEATURE: PLANE
NAME: face_12
DRF
TYPE : THREE_FACES
FACE : (face_1)
FACE : (face_2)
FACE : (face_3)
ED_SIZE
MAT_COND: NONE
TOLERANCE: NONE
ED_FORM
MAT_COND: NONE
TOLERANCE : NONE
TYPE : NONE
ED_POS
MAT_COND: M
TOLERANCE: CONSTANT
RADIUS: 1.0

DT_NODE
FEATURE: PLANE
NAME: face_6
DRF
TYPE : ONE_FACE
FACE : (face_1)
ED_SIZE
MAT_COND: NONE
TOLERANCE: NONE
ED_FORM
MAT_COND: M
TOLERANCE : CONSTANT
RADIUS: 1.0
TYPE: ANGULAR
ED_POS
MAT_COND: NONE
TOLERANCE: NONE

DT_NODE
FEATURE: PLANE
NAME: face_4
DRF
TYPE : THREE_FACES
FACE : (face_1)
FACE : (face_2)
FACE : (face_3)
ED_SIZE
MAT_COND: NONE
TOLERANCE: NONE
ED_FORM
MAT_COND: M
TOLERANCE : CONSTANT
RADIUS: 1.0
TYPE: SURF_PROFILE
ED_POS
MAT_COND: NONE
TOLERANCE: NONE

DT_NODE
FEATURE: PLANE
NAME: face_16
DRF
TYPE : THREE_FACES
FACE : (face_1)
FACE : (face_2)
FACE : (face_3)
ED_SIZE
MAT_COND: M
TOLERANCE: CONSTANT
RADIUS: 1.0
ED_FORM
MAT_COND: NONE
TOLERANCE : NONE
TYPE : NONE
ED_POS
MAT_COND: NONE
TOLERANCE: NONE

DT_NODE
FEATURE: PLANE
```

```
NAME: face_17
DRF
TYPE : THREE_FACES
FACE : (face_1)
FACE : (face_2)
FACE : (face_3)
ED_SIZE
MAT_COND: M
TOLERANCE: CONSTANT
RADIUS: 1.0
ED_FORM
MAT_COND: NONE
TOLERANCE : NONE
TYPE : NONE
ED_POS
MAT_COND: NONE
TOLERANCE: NONE

DT_NODE
FEATURE: LINE
NAME : edge_35
DRF
TYPE : THREE_FACES
FACE : (face_1)
FACE : (face_2)
FACE : (face_3)
ED_SIZE
MAT_COND: M
TOLERANCE: CONSTANT
RADIUS: 1.0
ED_FORM
MAT_COND: M
TOLERANCE : CONSTANT
RADIUS : 1.0
TYPE : CIRCULARITY
ED_POS
MAT_COND: M
TOLERANCE: CONSTANT
RADIUS : 1.0

DT_NODE
FEATURE: LINE
NAME: edge_44
DRF
TYPE : THREE_FACES
FACE : (face_1)
FACE : (face_2)
FACE : (face_3)
ED_SIZE
MAT_COND: M
TOLERANCE: CONSTANT
RADIUS: 1.0
ED_FORM
MAT_COND: M
TOLERANCE : CONSTANT
RADIUS : 1.0
TYPE : CIRCULARITY
ED_POS
MAT_COND: M
TOLERANCE: CONSTANT
RADIUS : 1.0

DT_NODE
FEATURE: PLANE
NAME: face_5
DRF
TYPE : THREE_FACES
FACE : (face_1)
FACE : (face_2)
FACE : (face_3)
ED_SIZE
MAT_COND: M
TOLERANCE: CONSTANT
RADIUS: 1.0
ED_FORM
MAT_COND: M
TOLERANCE : CONSTANT
RADIUS : 1.0
TYPE : FLATNESS
ED_POS
MAT_COND: M
TOLERANCE: CONSTANT
RADIUS : 1.0
```